FUNDAÇÃO DE ENSINO EURÍPIDES SOARES DA ROCHA
CENTRO UNIVERSITÁRIO EURÍPIDES DE MARÍLIA - UNIVEM
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

RODRIGO YOSHIO TAMAE

SISPRODIMEX – SISTEMA DE PROCESSAMENTO DISTRIBUÍDO DE IMAGENS MÉDICAS COM XML

MARÍLIA
2005

Livros Grátis

http://www.livrosgratis.com.br

Milhares de livros grátis para download.

RODRIGO YOSHIO TAMAE

SISPRODIMEX – SISTEMA DE PROCESSAMENTO DISTRIBUÍDO DE IMAGENS MÉDICAS COM XML

Dissertação apresentada ao Programa de Mestrado do Centro Universitário Eurípides Soares da Rocha, para a obtenção do Título de Mestre em Ciência da Computação.

Orientador:
Prof. Dr. Marcos Luiz Mucheroni

MARÍLIA
2005

TAMAE, Rodrigo Yoshio

SISPRODIMEX – Sistema de Processamento Distribuído de Imagens Médicas com XML / Rodrigo Yoshio Tamae; orientador: Marcos Luiz Mucheroni. Marilia, SP, 2005.

85 f.

Dissertação (Mestrado em Ciências da Computação) — Centro Universitário Eurípides de Marília – Fundação de Ensino Eurípides Soares da Rocha.

Introdução. 1.PACS e DICOM. 2. Tecnologias Java e XML. 3. Computação Paralela e Distribuída. 4. SISPRODIMEX – Um Sistema de Distribuição de Imagens Médicas. Conclusões. Referências.

CDD:

AGRADECIMENTOS

Ao meu pai (em memória) e à minha mãe que sempre me apoiaram em todas as etapas da minha vida.

À família do senhor Takayuki Oda que considero como sendo minha segunda família e que sempre fizeram muito por mim sem nada esperar em troca.

Ao meu orientador, em especial, um Grande Amigo, Professor Dr. Marcos Luiz Mucheroni, pelo apoio incondicional, pelo profissionalismo, pelo caráter, pela compreensão e que me motivou a chegar até aqui. Um homem admirável que materializa a essência de um verdadeiro Mestre: Jamais ser um doador da verdade, e sim, um apontador da verdade que o aluno tem que descobrir por si mesmo.

A minha grande amiga Sueli Yukie Oda que esteve sempre presente nos momentos mais difíceis desta e de outras jornadas.

A todos os professores com quem tive a felicidade de conviver e que muito enriqueceram a minha vida profissional.

Aos meus colegas do grupo de Arquitetura de Sistemas Computacionais, em especial, ao André Gobbi e Vasco Correia que colaboraram muito no desenvolvimento do projeto.

Aos meus amigos: Eduardo, Weber, Francis, Paulo, Miguel, José Júlio, Sormani e Ely.

TAMAE, Rodrigo Yoshio. **SISPRODIMEX – Sistema de Processamento Distribuído de Imagens Médicas com XML**. 2005. 85 f. Dissertação (Mestrado em Ciência da Computação) — Centro Universitário Eurípides de Marília – Fundação de Ensino Eurípides Soares da Rocha, 2005.

Resumo

O SISPRODIMEX (Sistema de Procesamento Distribuído de Imagens Médicas com XML) é uma plataforma cuja proposta é disponibilizar acesso a um banco de dados (imagens e informações médicas oriundas do padrão DICOM 3.0) através de tecnologia de ponta e de relativo baixo custo, como Java e XML, possibilitando que centros hospitalares possam contar com uma base centralizada de dados, permitindo acesso aos seus médicos e equipe devidamente credenciada a essa base independentemente de localização geográfica através de recursos da Internet e World Wide Web. O sistema conta ainda com um módulo de processamento distribuído baseado em *cluster*s (arquitetura Beowulf) visando minimizar o impacto nas solicitações de processamento.

Palavras-chave: PACS, DICOM 3.0, Java, XML, Web Semântica e Arquitetura distribuída.

TAMAE, Rodrigo Yoshio. **SISPRODIMEX – Sistema de Processamento Distribuído de Imagens Médicas com XML**. 2005. 85 f. Dissertação (Mestrado em Ciência da Computação) — Centro Universitário Eurípides de Marília – Fundação de Ensino Eurípides Soares da Rocha, 2005.

Abstract

SISPRODIMEX is the name attributed to a plataform whose purpose is supply access to a data-base (deriving image and medical information of DICOM standard) through high technology and neutral architecture with relatively low cost, like Java and XML, allowing that medical centers could be count on a centralized data-base allowing access to the doctors and trusted staff to this base independently of geographic localization through Internet and World Wide Web resources. The system still counts on a distributed processment system based in *cluster* (Beowulf Architecture) aiming at to reduce the impact in the processing requests.

Keywords: PACS, DICOM 3.0, Java, XML, Semantic Web e Distributed architecture.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

ACR	American College of Radiology
ANSI	American National Standard International
API	Application Programming Interface
CSS	Cascading Style Sheets
CT	Computed Tomography
DICOM 3.0	Digital Imaging and Communications in Medicine
DLL	Dynamic Link Libraries
DOM	Document Object Model
DTD	Data Description Language
HIS	Hospital Information System
HTML	Hiper Text Markup Language
IDE	Integrated Development Environment
IDL	Interface Definition Language
IIOP	Internet Inter-ORB Protocol
InCor	Instituto do Coração – Hospital das Clínicas/Faculdade de Medicina da Universidade de São Paulo (USP)
ISO	International Organization for Standardization
JDBC	Java Database Connectivity
JMPI-Plus	Java Message Passing Interface Plus
JVM	Java Virtual Machine
MPI	Message Passing Interface
MPICH	Message Passing Interface Chamaleon
MRI	Magnetic Ressonance Image
NEMA	National Electrical Manufactures Association
ODBC	Open Database Connectivity
OLIA	Ontologies with Language Intention and Agents
PACS	Picture Archiving and Communications Systems
PET	Positronic Emission Tomography

PVM	Parallel Virtual Machine
RDF	Recourse Definition Framework
RIS	Radiology Information System
RMI	Remote Method Invocation
RPC	Remote Procedure Call
SAX	Simple API for XML
SCAR	Society for Computer Applications in Radiology
SEDEC	Small Embedded Data Center named Cognitio
SGBD	Sistema Gerenciador de Banco de Dados
SGML	System Generalized Markup Language
SISPRODIMEX	Sistema de Processamento Distribuído de Imagens Médicas com XML
SOAP	Simple Object Access Protocol
SPECT	Single Photon Emission Computed Tomography
SQL	Structured Query Language
TCP-IP	Transmission Control Protocol – Internet Protocol
UDDI	Universal Description Discovery and Integration
URL	Uniform Resource Identifier
W3C	World Wide Web Consortium
WSDL	Web Service Description Language
XML	Extensible Markup Language
XSL	Extensible Stylesheet Language

SUMÁRIO

INTRODUÇÃO

O poder gerado pela disseminação da informação e pelo rápido crescimento das tecnologias de comunicação, tais como as redes de computadores e a Internet, tem afetado diretamente os mais diferentes ramos profissionais e científicos. E a área médica é uma das que mais tem colhido os benefícios deste desenvolvimento tecnológico.

Atualmente, os hospitais estão investindo fortemente em tecnologia, adquirindo equipamentos de última geração e informatizando seu atendimento aos pacientes, como na entrada de dados, geração de laudos e aquisição de imagens, obtendo, com isso, maior rapidez, qualidade de atendimento e reduzindo seus custos (HUANG, 1999).

As imagens médicas são amplamente utilizadas em instituições hospitalares como uma das principais ferramentas para subsidiar o diagnóstico na prática da medicina moderna, pois, têm oferecido um meio para monitorar os efeitos de tratamentos, planejar cirurgias e auxiliar o diagnóstico. No entanto, devido à grande utilização de equipamentos digitais de diagnóstico, uma instituição que mantém, por exemplo, um Centro ou Departamento de Radiologia gera uma enorme quantidade de dados sobre imagens médicas, o que torna crítica a realização das tarefas de armazenamento, distribuição e manipulação desses dados. Aliado a isto, está a incompatibilidade do formato das informações digitais geradas pelos diferentes sistemas de aquisições existentes no meio clínico (SANTOS, 2002).

Segundo relatório da Sociedade para Programas de Computador em Radiologia (*Society for Computer Applications in Radiology* - SCAR), é necessário uma mudança de paradigma no processo de interpretação radiológica, para ser possível lidar com a crescente quantidade de informação. O relatório aponta que as tarefas de visualização e processamento de imagens, combinadas com sistemas de assistência computacional, como sistemas de apoio

à decisão e acesso a bibliotecas de referências, estão entre os seis principais desafios para o futuro da pesquisa em imagens médicas (ANDRIOLI, 2003).

São nos chamados sistemas PACS (*Picture Archiving and Communications Systems*) que as imagens são, normalmente, geradas e armazenadas num padrão conhecido e amplamente adotado chamado DICOM 3.0.

O crescimento mundial de sistemas PACS e, conseqüentemente, do padrão DICOM, tem fortalecido o conceito de *Filmless Hospital*, uma vez que o tradicional sistema de filmes tem sido substituído, aos poucos, pelos modernos sistemas digitais de aquisição e armazenamento de imagens. Um processo bastante lento que, tanto no Brasil quanto no exterior, deve demorar algumas décadas para se concretizar.

Devido ao rápido avanço da Internet e a necessidade cada vez maior de criar mecanismos de interoperabilidade em ambientes PACS, devido às suas naturezas portáveis e distribuídas, as linguagens Java e XML emergem como uma opção para o desenvolvimento de aplicações médicas e clínicas com este fim.

Percebe-se claramente que a Internet tornou-se um dos principais veículos de comunicação utilizados mundialmente para acesso, recuperação e utilização de informações. No entanto, com o crescimento desordenado da Internet, a busca mais precisa por informações está se tornando ineficiente. Com o advento da Web Semântica, busca-se a possibilidade de obtenção de mecanismos mais eficientes de pesquisa.

Assim, o objetivo deste trabalho, através das tecnologias citadas, é propor uma plataforma com a finalidade de oferecer suporte a hospitais, clínicas e laboratórios de todos os portes (com independência de localização geográfica), acesso a um banco de dados contendo informações textuais e imagens médicas a partir de objetos DICOM sem o uso de sofisticados sistemas de *Workstations* (pelo contrário, garantindo isso através do uso de tecnologias de

relativo baixo custo e de alto desempenho), bem como, através de gerenciamento de contexto, otimizar, facilitar e organizar as informações geradas por ambientes PACS.

Esta plataforma é denominada de SISPRODIMEX (Sistema de Processamento Distribuído de Imagens Médicas com XML) e conta com um sistema de *Data Warehouse* conectado a um sistema experimental de *cluster* para obtenção de performance baseado na arquitetura Beowulf.

Assim, uma vez conectado ao SISPRODIMEX, o usuário poderá efetuar a requisição de todos os serviços disponibilizados pela plataforma a partir de qualquer localidade através de um navegador Internet. O poder de processamento da parte cliente será somente exigido para o tratamento de respostas geradas em XML, HTML ou apenas para visualizar a imagem, tornando-o ágil e simples.

Este trabalho está organizado da seguinte forma: no capítulo 2 serão abordados os conceitos de ambientes PACS e sua relação com a tecnologia de objetos DICOM. No capítulo 3 serão abordados alguns aspectos essenciais de Java e XML. O capítulo 4 apresentará os conceitos de Computação Paralela e Distribuída. No capítulo 5 será fornecida uma descrição sobre a plataforma SISPRODIMEX. No capítulo 6 serão apresentadas as conclusões deste trabalho e propostas de trabalhos futuros e no capítulo 7 são apresentadas as referências bibliográficas.

CAPÍTULO 1 – PACS e DICOM

1.1 Introdução

As atividades médicas são marcadas por uma busca constante de um diagnóstico preciso e da avaliação da terapêutica. Para esse fim o médico serve-se de uma grande variedade de técnicas de produção de imagens, entre as quais destacam-se os métodos radiológicos (Tomografia Computadorizada (*Computed Tomography* - CT), a Ressonância Magnética (*Magnetic Ressonance Image* - MRI), o Ultra-som, a Tomografia de Emissão Positrônica (*Positronic Emission Tomography* - PET), a Tomografia Computadorizada por Emissão de Fótons (*Single Photon Emission Computed Tomography* – SPECT) e a Medicina Nuclear (*Nuclear Medicine*). A mais tradicional das modalidades é o Raio-X). Esses métodos são chamados de modalidades de imagem médica (SIEGEL, 1999A).

Devido à relutância de médicos e administradores hospitalares mais tradicionalistas na adoção dos computadores, a área hospitalar estava, de certa forma, atrasada, mas a necessidade de manejo e manipulação das informações em larga escala e com altas velocidades os venceram. O trabalho com imagens médicas, por intermédio do uso de computadores, pode ser dividido em quatro categorias: geração de imagens, análise de imagens, gerenciamento de imagens e gerenciamento de informações (GREENES, 1990).

Embora não sendo parte integrante essencial deste trabalho, o Processamento Digital de Imagens deve ser considerado, nestes ambientes, um aspecto extremamente importante, uma vez que seus algoritmos requerem, na maioria das vezes, alto desempenho e funcionalidade para permitir o tratamento adequado das imagens coletadas, e embora não seja

aspecto central deste trabalho, é feita uma pequena introdução a estes conceitos, visando estabelecer uma ligação inicial do SISPRODIMEX, com trabalhos futuros nesta linha.

1.2 Surgimento do Processamento de Imagens Médicas e a Evolução da Radiologia

Em 1895, o físico alemão Wilhelm Conrad Röntgen descobriu os raios-X, que revolucionaram o meio científico, e em especial a Medicina, de tal forma que por volta de 1900 a radiologia já existia como especialidade médica (NOBEL, 2003). Por volta de 1940 novas tecnologias como a de intensificadores de imagens permitiram a realização de fluoroscopias de ótima qualidade e em tempo real, as quais foram os únicos métodos existentes até a década de 70. O desejo de separar estruturas superpostas também levou ao desenvolvimento de uma variedade de técnicas tomográficas analógicas, especialmente a tomografia axial (seções transversais), mas que davam maus resultados. Os pesquisadores reconheceram, então, que um computador seria necessário para realizar a limpeza dos borrões, e métodos matemáticos para reconstrução de imagens foram desenvolvidos, principalmente por Cormack (CORMACK, 1979). Por volta de 1970, Hounsfield e sua equipe da EMI Corporation desenvolveram o primeiro tomógrafo computadorizado comercialmente viável, que permitiu a visualização de estruturas internas do corpo através de seções transversais, trabalho pelo qual ambos os pesquisadores receberem o prêmio Nobel de Medicina em 1979 (HOUNSFIELD, 1979).

Após a invenção do tomógrafo computadorizado, vários métodos de produção de imagens foram desenvolvidos, como a MRI, que produz cortes tomográficos a partir de campos magnéticos, a ultra-sonografia, e a cintilografia, com o uso de isótopos radioativos

que além de gerar imagens de estruturas anatômicas, presta-se à avaliação da função orgânica; e entre as quais se conta o SPECT e o PET.

Pode-se atribuir a muitos fatores a multiplicação das modalidades de produção de imagens médicas, tais como a melhor compreensão dos princípios básicos da captação de imagens, aperfeiçoamento de técnicas matemáticas de reconstrução, a evolução dos computadores com desenvolvimento de equipamentos mais baratos e mais seguros. Esta melhoria na tecnologia da computação levou a uma tendência para a geração de imagens digitais e os exames tradicionais de raios-X puderam ser adquiridos e processados pelo computador. A aquisição e análise de imagens digitais de raios-X formam a base de um novo campo chamado Radiologia Digital.

Estes sistemas foram concebidos para a área médica e denominados de sistemas PACS.

1.3 Definindo os sistemas PACS

Os chamados sistemas PACS são sistemas de computadores usados para a aquisição, armazenamento, processamento, exibição e distribuição de imagens médicas em formato digital. Esses sistemas têm como objetivo disponibilizar a imagem para diagnóstico e permitir o acesso dessas imagens médicas a partir de uma rede ou localidade remota (SIEGEL, 1999A).

Grandes fornecedores (*players*) estão, atualmente, envolvidos no desenvolvimento e no fornecimento da tecnologia PACS como Fuji film, Kodak, AGFA, Siemens, GE, Toshiba, Phillips, Sun, IBM, Compaq e outros (WHITE-SAND, 2003).

Para demonstrar a importância dos sistemas de processamento de imagens médicas, pode-se citar, como exemplo, as soluções adotadas no sistema PACS do InCor (Instituto do Coração - HC/FMUSP) para o armazenamento de imagens médicas e as soluções para a

distribuição destas imagens a todo o hospital utilizando o Prontuário Eletrônico do Paciente baseado em Web (BERTOZZO, 2003).

A Figura 1 mostra a estrutura do sistema PACS adotada pelo InCor em que é possível observar, principalmente, os equipamentos que representam as modalidades médicas, bem como o uso do sistema de interconexão baseado na topologia Ethernet, uma vez que o protocolo utilizado para comunicação neste sistema PACS é o TCP-IP. Este protocolo foi escolhido, também, com a finalidade de garantir a interoperabilidade de equipamentos independentemente da localização geográfica. É possível, também, observar um equipamento denominado 'Conversor DICOM' cuja finalidade é converter imagens geradas no padrão analógico em formato digital padrão DICOM.

Como se pode observar, a tecnologia PACS pode ser utilizada para obter operações quase sem filme ou totalmente sem filme (*filmless*). Pode-se denominar "*filmless*" um hospital com um ambiente de rede amplo e integrado, no qual o filme foi completamente (ou em grande parte) substituído por sistemas eletrônicos que adquirem, arquivam, disponibilizam e exibem imagens (SIEGEL, 1999A) (MARQUES, 2000).

O PACS em conjunto com os Sistemas de Informação em Radiologia (RIS) e de Informação Hospitalar (HIS) formam a base para um serviço de radiologia *filmless* (SIEGEL, 1999B) (MARQUES, 2000).

A implantação de um serviço de radiologia sem filme deverá trazer melhorias no que se refere à acessibilidade e integração de informações, pela vinculação de imagens ao registro médico eletrônico do paciente, e no que se refere à aplicação de novas técnicas e desenvolvimento na aquisição, exibição e processamento de imagens (SIEGEL, 1999B) (MARQUES, 2000).

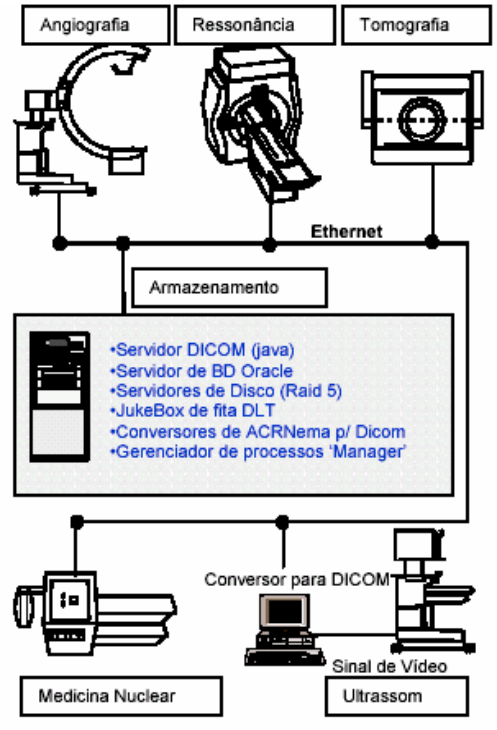

Figura 1 – Estrutura PACS do InCor

Um alto nível de integração do PACS na operação dos dados é necessário para garantir sua ótima funcionalidade, o que requer uma quantidade enorme de planejamento e implementação com analistas de sistemas, engenheiros, a administração e também com os departamentos clínicos. Isso é muito mais complexo de se obter do que qualquer outro processo de aquisição para o departamento de radiologia e o hospital (SIEGEL, 1999B).

Um outro exemplo, o Royal Glamorgan Hospital, de South Wales, no Reino Unido, que atende a uma população local de 280.000 habitantes, adotou um sistema PACS classificado como *filmless system*, que permite a visualização de uma imagem em uma workstation ao invés de analisar um exame de raios-X em uma caixa de visualização tradicional. Além disso, estas imagens geradas por computador podem, se necessário, ser impressas para uso fora das dependências do hospital (SHIRALKAR, 2003).

1.4 Padrão DICOM 3.0

O DICOM 3.0 é o formato padrão para aquisição de imagens médicas e informações textuais relacionadas, e que garante a total fidelidade da imagem em formato digital, adquirida pelo equipamento da modalidade (SIEGEL, 1999A).

O DICOM descreve os mecanismos necessários para a troca de informações entre os diversos tipos de equipamentos médicos. Isto significa dizer que o modelo descreve como os fabricantes devem implementar o padrão DICOM em seus equipamentos com a finalidade de garantir que eles possam se comunicar com outros equipamentos do mesmo padrão. Este mecanismo é chamado DICOM Conformance Statements e cada modelo de equipamento tem o seu próprio (WHITE-SAND, 2003).

O padrão foi desenvolvido por um comitê formado por membros do *American College of Radiology* (ACR) e do *National Electrical Manufactures Association* (NEMA) que iniciou os trabalhos em 1983. Este comitê foi formado com o intuito de desenvolver um padrão para comunicação digital e informações de imagens médicas. O comitê publicou a primeira versão do padrão em 1985 e foi chamado de ACR-NEMA 300-1985 ou ACR-NEMA Version 1.0, e a segunda versão em 1988, chamada de ACR-NEMA 300-1988 ou ACR-NEMA Version 2.0. A terceira versão do padrão foi denominada de DICOM 3.0 e foi apresentada em 1993 quando o conteúdo foi alterado, foram discutidos alguns problemas e implementado o protocolo TCP/IP para comunicação em rede (WHITE-SAND, 2003).

O DICOM utiliza os conceitos que são comuns ao mundo da linguagem de programação orientada a objetos (POO). Isto porque a cada imagem adquirida, ela é instanciada a partir de um modelo (classe) transformando-se em um conjunto de informações com características particulares e únicas que serão, então, armazenadas (objeto) (WHITE-SAND, 2003). Devido a isso é correto dizer objeto DICOM e não imagem DICOM.

Os equipamentos de modalidade geram imagens em formato baseado em pixels. Um dos mais utilizados é o JPEG. Outros formatos, porém, também são suportados como o JPEG2000, Tiff e outros (WHITE-SAND, 2003).

Segue as regras de normalização ANSI (*American National Standards Institute*) e caracteriza-se por ser um padrão de adesão e não de imposição, ou seja, usar ou não as características do padrão DICOM fica a critério do fornecedor e isto não lhe será cobrado. Isso depende do quanto o fornecedor quer manter compatível o seu equipamento com os demais existentes no mercado.

São os objetivos primários do padrão (WHITE-SAND, 2003):

- Promover a comunicação de informações e imagens em formato digital independentemente dos fabricantes dos equipamentos.
- Permitir a criação e expansão de sistemas PACS entre os diversos sistemas de informação hospitalar.
- Permitir a criação de uma base centralizada de informações médicas com independência de localização geográfica.
- Endereçar a semântica de comandos e dados associados. Para que equipamentos possam atuar uns sobre os outros, deve haver padrões de modo que equipamentos estejam esperando para reagir a comandos e dados associados.
- É explícito em determinar a adaptação necessária de implementação do padrão. Em particular, uma instrução de adaptação deve especificar informações suficientes a fim de que a necessidade de trabalho conjunto possa ser prevista com outro equipamento que também se ajuste ao padrão.
- Facilitar operações em ambientes de rede, sem a necessidade de um mecanismo de interface de rede.

- É estruturado para acomodar a introdução de novos serviços, facilitando assim suporte para futuras aplicações em imagens médicas.
- Faz uso de padrões internacionais existentes sempre que aplicável e adequou-se à documentação estabelecida para padrões internacionais.

O padrão DICOM facilita a capacidade de atividade conjunta de equipamentos de imagens médicas porque especifica:

- Um conjunto de protocolos a serem obedecidos pelos equipamentos exigindo a adaptação deles ao padrão.
- A sintaxe e semântica de comandos e informações associadas, as quais devem ser trocadas usando estes protocolos.
- Informações que devem ser fornecidas com uma implementação para que a adaptação para o padrão seja cumprida.

O padrão DICOM, no entanto, não especifica:

- A implementação detalhada de algumas características do padrão sobre um equipamento que exige adaptação.
- O conjunto completo de características e funções esperadas de um sistema implementado integrando um grupo de equipamentos, cada um exigindo adaptação DICOM.
- Um processo de teste e validação para avaliar uma adaptação ao padrão.

O padrão DICOM apresenta, porém, alguns problemas. É importante lembrar que devido ao uso de imagens baseadas em pixels, as imagens geradas são de grande volume e exigem dispositivos de grande poder de armazenamento (BERTOZZO, 2003).

Objetos DICOM oriundos de equipamentos de Angiografia Digital por Raios-X têm um tamanho elevado, especialmente, quando são adquiridas para análise de crianças, na qual é utilizada uma taxa de aquisição de 30 imagens por segundo. Seu tamanho pode chegar

facilmente a 512 linhas X 512 colunas X 300 *frames*, ou seja, aproximadamente 75Mbytes, sendo que cada estudo contém cerca de 10 imagens, então, tem-se aproximadamente 750Mbytes (MORENO, 2004).

O nível de compactação de imagem aceita pelo padrão é muito pequeno, raramente superior a trinta por cento (*lossy*) para eliminar a possibilidade de perdas na imagem (WHITE-SAND, 2003). Normalmente, as imagens são armazenadas sem nenhuma taxa de compactação (*lossless*) e, de acordo com o equipamento e com o fabricante, não existe a opção de compactação.

Este gargalo gera problemas não só quanto ao armazenamento, mas principalmente quanto à transferência de estudos via rede, uma vez que, um dos objetivos é permitir o acesso às imagens a partir de localidades remotas.

Por ser uma tecnologia recente e enraizada em um nicho de mercado muito forte, a tecnologia que envolve o padrão DICOM vem sendo foco de muitos estudos. A maioria destes estudos e propostas sugerem meios alternativos para solucionar ou amenizar as deficiências que envolvem o uso do padrão DICOM.

Entretanto, tecnologias como as linguagens Java, XML e arquiteturas de computação paralela e distribuída baseadas em *clusters* permitem a construção de soluções que podem contribuir substancialmente para amenizar tais deficiências.

CAPÍTULO 2 – TECNOLOGIAS JAVA E XML

2.1 Introdução

A portabilidade e a realativa facilidade de desenvolvimento de aplicações Web fazem de Java e XML uma boa opção para o desenvolvimento e implementação de aplicações médicas e clínicas, considerando a heterogeneidade e a natureza distribuída do ambiente clínico.

A linguagem Java permite aos programadores criarem páginas Web com conteúdo dinâmico e interativo, desenvolver aplicativos em larga escala, aprimorar a funcionalidade de servidores WEB, fornecer dispositivos para aplicativos de consumidor e muito mais (DEITEL, 2003A).

As aplicações em Java são também mais robustas que as aplicações correspondentes em C ou C++, porque um sistema *run-time* Java ou JVM (*Java Virtual Machine*), que geralmente acompanha os navegadores, gerencia toda a memória. As mesmas características que fornecem robustez também fornecem segurança, mesmo quando as aplicações trafegam pela Internet, pois, a JVM possui mecanismos de segurança que protegem contra interferências indevidas. Aplicações que executam tarefas concorrentes são mais rápidas, pois, a linguagem Java possui suporte próprio para tratamento de concorrência (HAMILTON, 1996) (PERRY, 1996).

Devido às transformações ocorridas desde o surgimento da *World Wide Web* (www), hoje, existe o XML (*eXtensible Markup Language*), a primeira linguagem que torna os documentos legíveis para as pessoas e manipuláveis por computadores, um resultado obtido por meio de um conjunto poderoso de marcas (*tags*) que têm uma sintaxe mais bem definida,

flexível e extensível que o de HTML (*Hiper Text Markup Language*). É a linguagem do documento inteligente, um passo à frente em relação aos métodos convencionais de representação de documentos que dependem do formato, e não da estrutura (DEITEL, 2003C).

Embora parecida semanticamente com HTML, ela tem inúmeras possibilidades de criação de novas de *tags*, de acordo com o grupo semântico e área de documentação, e isto a torna ainda mais poderosa e flexível, enquanto ela própria torna-se uma metalinguagem.

A independência dos dados, a separação do conteúdo e sua apresentação são as características essenciais de XML. Uma vez que um documento XML descreve dados, ele pode ser processado por um aplicativo. A ausência de instruções de formatação facilita a realização de análise sintática. Isso torna XML uma estrutura de referência que pode ser usada para o intercâmbio de dados (DEITEL, 2003C).

Na área médica, um sistema *Electronic Health Record* (EHR) é definido como o armazenamento digital de informações médicas de indivíduos com o propósito de dar suporte contínuo de tratamentos, saúde e educação. Um EHR pode incluir desde simples observações a testes laboratoriais, imagens médicas, tratamentos, terapias, administração de drogas, informações de identificação do paciente, permissões legais e muito mais. Sistemas EHR poderão contribuir para tratamentos mais efetivos e eficientes por facilitar a unificação de informações clínicas a partir de uma grande quantidade de pacientes de inúmeras localidades. Atualmente, as informações clínicas de um paciente geralmente encontram-se armazenadas em pontos isolados e nunca disponíveis (COHEN, 2004).

Tudo isso gera uma enorme quantidade de dados médicos e a maior parte deles não são padronizados. Acredita-se que agregar estes dados seja tão importante que no futuro da medicina serão incluídos nos sistemas EHR. Tais sistemas irão coletar e converter dados a partir destas fontes dispersas em XML. Esta visão é extremamente importante pelo fato de que organizações internacionais estão prontas para desenvolver padrões XML para domínios

da área da saúde. Os sistemas EHR irão incluir mais mecanismos de indexação extensiva e busca que a própria Web. Muitas instituições irão utilizar sistemas EHR para executar operações de *data mining* e análise do montante de dados médicos (COHEN, 2004).

2.2 Aspectos relevantes da Linguagem Java

Java é uma linguagem de programação orientada a objetos com uma sintaxe semelhante ao C e C++, embora mais simples. Por ser uma linguagem interpretada, todo o ciclo de vida para o desenvolvimento de software pode ser realizada dentro de um navegador Internet.

A principal vantagem em Java é que as aplicações são completamente portáveis. Uma vez que o código é escrito não há necessidade de adaptá-lo ou mesmo recompilá-lo. Para um programa qualquer executar em um computador, ele deve primeiro ser traduzido de uma linguagem como Pascal ou C++ para a língua nativa da máquina, o que é feito pelo compilador. Como, geralmente, os softwares já vêm compilados, podem ser criadas diferentes versões para plataformas diferentes. Ao invés de produzir instruções específicas para máquinas distintas, o compilador Java produz o bytecode independente de plataforma. O ambiente *run-time* Java ou máquina virtual traduz o bytecode (código intermediário gerado a partir da compilação de código-fonte em linguagem Java) em instruções específicas para a máquina em uso. A máquina virtual Java é instalada na máquina do usuário, ou como parte de um navegador ou como parte do sistema operacional (HAMILTON, 1996) (PERRY, 1996).

Como a maioria dos programas orientados a objetos, todas as variáveis e funcionalidades em Java estão contidas dentro de objetos que, por sua vez, são definidas através de classes. Como C, Java contem tipos de dados numéricos padrões, porém, independentes de hardware ou sistema. Em Java, todas as variáveis são fortemente tipadas e

não há ponteiros. Isto produz uma redução de erros de programação e um aumento de segurança (HAMILTON, 1996). As aplicações Java são livres de virus, pois, não podem acessar a memória do sistema como os programas em C ou C++ (PERRY, 1996).

Java é uma linguagem que tem um grande conjunto de qualidades de modo que podem ser utilizadas em diversas aplicações. Dentre elas, pode-se citar: interligação com sistemas CORBA (JavaIDL), suporte ao desenvolvimento de servidores (servlets), gerenciamento (JMAPI), internacionalização, independência de plataforma (oferecendo suporte desde *smartcards* até *mainframes*), acesso a base de dados (JDBC), objetos distribuídos (JavaRMI), comércio eletrônico (JavaCommerce), componentização (JavaBeans), multimídia (JavaMedia), telefonia (JavaTelephony), suporte à voz (JavaSpeech), segurança (JavaSecurity) entre outras.

Devido a esta crescente lista de capacidades, considerou-se Java como sendo a linguagem ideal para o desenvolvimento deste projeto devido a suas características multiplataforma e de relativo baixo custo.

2.3 Alguns conceitos fundamentais de XML

A XML é uma tecnologia aberta e amplamente suportada para a troca de dados. Tem como principal característica a possibilidade de permitir a criação de marcas (*tags*) personalizadas, possibilitando ao autor do documento XML construir o modo como as informações serão estruturalmente representadas (DEITEL, 2003B). Como tem suporte ao formato ASCII e *Unicode* permite a criação de documentos em texto puro e em praticamente qualquer tipo de forma estruturada. Por ser estruturalmente mais leve que a HTML permite que seja suportada por equipamentos de qualquer porte.

Foi desenvolvida pelo *World Wide Web Consortion* (W3C) com a finalidade de torná-lo um formato padrão para troca de dados na Internet (W3C-XML, 2004). Apesar de ter sido desenvolvida como uma tecnologia para Internet, a XML também pode ser utilizada em qualquer tipo de aplicação multicamada.

Tendo surgido como uma simplificação do SGML (*Standard Generalized Markup Language*), a primeira linguagem baseada em marcadores, o XML permite a construção de estruturas chamadas de Documentos XML. Um Documento XML possui um conjunto de marcadores que são definidos por uma gramática própria denominada DTD (*Document Type Definition*). Estes marcadores especificam a estrutura lógica do documento e os dados por eles delimitados são chamados de entidades. É o DTD quem define a validação de um Documento XML e este é considerado válido somente se sua estrutura conter as entidades na ordem, quantidade e aninhamento definidas no DTD (CORREIA, 2004).

O XML se tornou muito popular em conseqüência de sua simplicidade e robustez. Quando as aplicações começavam a se tornar mais complexas e distribuídas foram criados diversos protocolos e formatos de arquivos, o que dificultava a compatibilidade e interoperabilidade entre elas. O XML resolveu estes problemas de forma padronizada e permitiu a consistência de informações. Como prova da força que o XML vem ganhando, pode-se citar o fato que vários fornecedores (alguns deles, poderosos nomes como IBM, Sun Microsystems e outros) tem disponibilizado processadores XML que oferecem recursos para construção, validação e processamento de um Documento XML.

Por ser a XML uma solução para compartilhamento de dados é, naturalmente, uma opção para gravação ou transmissão de dados que possam ser representados como texto (como web sites, texto formatado, figuras vetoriais, planilhas, tabelas de dados, instruções de processamento e outros).

Usar o XML traz uma série de vantagens, como:

- Separação da estrutura da apresentação
- Informação semântica
- Independência de plataforma
- Facilidade de geração de visões diferentes de dados
- Facilidade de leitura por pessoas e máquinas
- Facilidade de compartilhamento de dados entre aplicações

No entanto, não é um bom formato para representar dados como imagens de bitmaps, vídeos, som e outros formatos multimídia em geral por serem grandes e geralmente armazenar quantidades de informação compactada.

Um documento XML descreve exclusivamente a estrutura dos dados. A forma como a informação será apresentada não importa. Assim, a XML força a separação entre os dados, seu significado e como devem ser expostos, tornando mais fácil a sua utilização em meios diferentes. O usuário ou o programa que receber o Documento XML decidirá o que fazer com ele.

Neste trabalho, o uso da XML justifica-se pela necessidade de troca de informações textuais sobre imagens médicas independentemente de plataforma através da Web.

2.4 Conexão Java e XML

A informação armazenada em documentos XML deve, em algum momento, ser lida por um programa para executar uma determinada função, como visualizar, modificar ou imprimir. A linguagem de programação Java entra como um fator de essencial importância no que diz respeito à conversão, por exemplo, de um documento XML em um documento HTML. Através da implementação de um *parser* (programa que serve para análise gramatical de uma determinada estrutura) nesta linguagem, é possível recuperar os elementos dos arquivos DTD

e XML, aplicar as estruturas e os estilos de acordo com a XSL e a CSS, gerando finalmente um documento HTML, que contém o resultado que pode ser apresentado no navegador.

Apesar de enfatizar-se a linguagem Java, qualquer outra linguagem de programação pode ser utilizada para implementação de um parser XML. Isto é possível devido, em grande parte, à existência de duas APIs (Application Programming Interface): SAX (*Simple API for XML*) e DOM (*Document Object Model*). Elas foram criadas com o propósito de permitir aos programadores acessarem a informação armazenada em documentos XML sem precisar escrever um parser em uma linguagem de programação específica. Assim, se a informação do documento for mantida no formato XML Versão 1.0 e se uma dessas duas APIs for utilizada, a aplicação pode usar qualquer parser XML disponível (IDRIS, 1999).

O poder de XML está na sua capacidade de representar a estrutura da informação baseada em como cada parte dessa informação se relaciona com as demais (BRAY,2000). Os documentos estruturados têm a propriedade de poderem ser aninhados um dentro do outro, apresentando uma estrutura em forma de árvore.

DOM (*Document Object Model*) fornece uma interface independente de plataforma e linguagem para a estrutura e o conteúdo de documentos HTML e XML.

Ele descreve uma linguagem neutra capaz de representar qualquer documento HTML ou XML em forma de uma árvore e tratar a informação armazenada nesses documentos como um modelo de objetos hierárquicos (IDRIS, 1999). DOM cria uma árvore de nós, baseada na estrutura e na informação do documento, sendo que o acesso à informação pode ser feito através de interações com essa árvore.

DOM busca fornecer um modelo padrão único na forma como são organizados os diversos objetos que formam os documentos. Ele também busca padronizar uma interface desses objetos para facilitar na navegação em documentos e no processamento deles. Além

disso, preserva a seqüência dos elementos lidos a partir dos documentos HTML e XML, como se fosse um documento.

Os principais objetivos da API DOM são (MCCARTH, 1999):

- Fornecer um conjunto de objetos e interfaces suficientes para representar o conteúdo e a estrutura dos documentos HTML e XML sem perda de informações significativas.
- Realizar isso de uma forma independente de plataforma e linguagem.
- Fornecer funcionalidade de criação de objeto poderosa o suficiente para permitir que documentos HTML e XML sejam criados completamente a partir do zero.
- Fornecer uma base sólida e extensível na qual podem ser adicionadas camadas DOM no futuro.

Diferente de SAX, em DOM o parser faz quase tudo: lê o documento, cria um modelo de objeto em uma linguagem e fornece uma referência a este modelo de objeto para poder controlá-lo (IDRIS, 1999).

Com o uso de DOM, os usuários se beneficiam de uma interface homogênea tanto para a HTML como para a XML. Eles também podem mover suas aplicações para qualquer plataforma compatível com DOM sem ter que efetuar qualquer alteração de código. E ainda é possível fazer a transformação de um documento XML para HTML, por exemplo, utilizando as especificações XSL.

A linguagem Java pode efetuar a conversão de um documento XML em uma estrutura de dados em árvore ou em uma seqüência ordenada de eventos. Neste caso, SAX e DOM são recursos que podem ser utilizados, pois existem bibliotecas que utilizam essas APIs em suas implementações e que disponibilizam vários métodos para realizar diversas

manipulações sobre os documentos XML. Assim, essas APIs auxiliam o programador a manipular e a apresentar documentos XML em suas aplicações.

A API DOM pode ser utilizada de modo mais abrangente do que a API SAX, pois a SAX não é muito adequada em atividades mais elaboradas em que é necessário modificar a estrutura do documento de forma mais complicada. Já DOM possui diversos métodos para realizar manipulações sobre a estrutura e o conteúdo do documento, permitindo que programas possam adicionar, modificar ou apagar nós da árvore de objetos a ele associados.

Existe, em Java, uma série de interfaces necessárias que estão definidas de acordo com a recomendação DOM, cada uma representando um determinado objeto da árvore de nós. As interfaces possuem diversos métodos que relacionam uma interface com a outra, podendo obter diversos tipos de informações sobre cada objeto da árvore (VELOSO, 2003).

2.5 Java RMI

Sendo uma linguagem de propósitos gerais, Java pode ser utilizada tanto para implementações para o lado cliente (applets) quanto para o lado servidor (servlets). Para tal abordagem, a Sun Microsystems definiu uma interface para a comunicação entre o servidor e o cliente baseados em Java chamada RMI (*Remote Method Invocation*).

RMI é um conjunto de classes Java, ferramentas de desenvolvimento e outras características que permitem a clientes Java executar métodos em objetos Java que estejam em diferentes executáveis Java (DEITEL, 2003B).

É importante observar que o RMI não suporta objetos distribuídos independentes de linguagem. O software utilizado tanto no cliente quanto no servidor deve ser implementado em Java, executando dentro de JVMs (DEITEL, 2003B).

A RMI é a implementação da RPC (*Remote Call Procedure)* por Java para comunicação distribuída de um objeto Java com outro. Uma vez que um método (ou serviço) de um objeto Java é registrado como sendo remotamente acessível, um cliente pode pesquisar (*lookup*) esse serviço e receber uma referência que permita ao cliente utilizar este serviço, isto é, chamar o método. A sintaxe da chamada de método é idêntica àquela de uma chamada para um método de outro objeto no mesmo programa. Como no RPC, a ordenação (*marshalling*) é tratada pela RMI. Entretanto, a RMI oferece transferência de objetos de tipos de dados complexos via o mecanismo de serialização de objeto. A classe *ObjectOutputStream* converte qualquer objeto Serializable em um fluxo de bytes que pode ser transmitido através de uma rede. A classe *ObjectInputStream* reconstrói o objeto original para utilizar no método receptor. O programador não precisa se preocupar com a transmissão dos dados sobre a rede. A RMI não exige do programador aprender uma IDL (*Interface Definition Language*) porque todo o código de rede é gerado diretamente a partir das classes existentes no programa. Além disso, uma vez que a RMI suporta somente uma linguagem, Java, nenhuma IDL neutra com relação a linguagem é requerida; as próprias interfaces de Java são suficientes (DEITEL, 2003B).

2.6 Acesso a Banco de Dados com Java

Um gerenciador de banco de dados possui um conjunto de bibliotecas, as DLLs (*Dynamic Link Libraries*), que permitem a conexão e o uso de aplicações. O acesso direto a estas bibliotecas é denominado de Chamadas Nativas.

O uso de chamadas nativas por um programa requer um conjunto de operações específicas ao banco de dados utilizado, além das fornecidas pelo compilador de linguagem. Por exemplo, pode-se dizer que não é possível usar o recurso de chamadas nativas para um banco de dados Oracle em um banco de dados DB2.

Há uma alternativa para as chamadas nativas denominadas ODBC (*Open Database Connectivity*), que é uma API (*Application Programming Interface*) para o acesso a banco de dados. A principal vantagem do ODBC sobre chamadas nativas convencionais é a criação de programas independentes do banco de dados utilizado. O ODBC possibilita a gerenciador de dispositivos controlar a comunicação entre a aplicação e o gerenciador de banco de dados. Assim, cada gerenciador de banco de dados registra o seu próprio dispositivo ODBC através deste gerenciador de dispositivo, o qual, após receber uma chamada ODBC de uma aplicação, interpreta a solicitação traduzindo-as em chamadas nativas ao gerenciador de banco de dados utilizado.

Com o intuito de utilizar os recursos da linguagem Java, a Sun Microsystems especificou um padrão para acesso a banco de dados por programas implementados em Java chamado JDBC (*Java DataBase Connectivity*). Como o ODBC, o JDBC é uma API que determina como os programas Java devem interagir com os bancos de dados. A principal diferença entre estes dois padrões é que o JDBC é inspirado em uma linguagem orientada a objetos, assim, em vez de especificar um conjunto de funções, o JDBC administra um conjunto de objetos e interfaces Java.

Para o desenvolvimento de sistemas direcionados a Internet que envolvam um modelo de armazenamento baseado em banco de dados relacional, é muito mais fácil utilizar formulários HTML como um front-end do que portar a aplicação para múltiplas plataformas. Porém, uma abordagem convencional, utilizando CGI com chamadas nativas ao banco de dados, pode apresentar alguns problemas. O HTML não incorpora nenhum recurso para validação dos dados de entrada. A validação de formulários pode ser feita somente no servidor. Isto significa que erros na entrada de dados não podem ser corrigidos até que o formulário seja processado pelo servidor Web, levando a interfaces não-interativas.

Outro problema neste tipo de abordagem é o consumo significativo, dependendo da complexidade do formulário e quantidade de usuários simultâneos, dos recursos do servidor para se executar a validação do formulário. Porém, o problema mais sério é a propensão para gargalos no servidor.

Uma API JDBC permite que uma aplicação se comunique diretamente com um sistema gerenciador de banco de dados (SGDB), eliminando a maioria dos problemas de um sistema convencional para a Web. Neste caso, o navegador carrega um applet, o usuário entra com os dados e o applet valida os campos preenchidos localmente antes que os dados sejam enviados para o SGDB via JDBC.

Para dar funcionalidade em sistemas Java, criar processos e tarefas, e permitir a manutenção em ambientes de redes, a integração de objetos em componentes e o crescimento das aplicações em rede, foram criados serviços em diversas linguagens que hoje estão disponíveis na Web e denominados Serviços Web ou Web Services.

2.7 Web Services

Web Services são conjuntos de aplicações autodescritivas que podem ser publicadas, localizadas e invocadas através da Web. Estas aplicações podem ser desde simples processos, como troca de mensagens, até complexas transações industriais, como a compra de mercadorias. Uma vez que um Web Service é publicado, outras aplicações (ou outros Web Services) podem acessá-lo e invocá-lo, tanto para a obtenção de dados como interação com serviços que uma organização oferece.

Diferente de outras tecnologias, Web Services não são acessados através de protocolos específicos de modelagem de objetos, como IIOP, RMI ou DCOM. São acessados através de protocolos e formatos de dados independentes de plataforma como HTTP, XML e

SOAP. A interface de um Web Service é acessível através de uma mensagem XML padronizada. São descritos utilizando um padrão formal chamado descrição de serviço, que envolve os detalhes necessários para a interação com o serviço, incluindo o formato das mensagens, tipos de dados e localização. A interface encapsula os detalhes de implementação do serviço, permitindo a sua utilização independente de plataforma de hardware ou software na qual está implementado o serviço (KREGER, 2001).

A Figura 2 ilustra a arquitetura dos Web Services baseada na interação de três categorias: provedor do serviço (*service provider*), provedor de registro (*registry provider ou registry broker*) e o cliente do serviço (*service requestor*). Estas categorias envolvem as operações de procura (*find*), publicação (*publish*) e acoplamento (*bind*) do serviço.O provedor publica o serviço com a operação *publish*. O cliente do serviço utiliza a operação *find* para obter uma descrição de serviço do provedor de registro e usa esta descrição para, dinamicamente, acessar e interagir (*bind*) com o provedor do serviço.

A interação entre Web Services pode ser feita estaticamente ou dinamicamente em tempo de execução. Um solicitante descreve as características do serviço procurado e utiliza o provedor de registro para localizar um serviço apropriado. Uma vez localizado o serviço, a informação na descrição do serviço é utilizada para a interação entre cliente e servidor. A descoberta, a invocação dinâmica de serviços (*publish, find, bind*) e uma colaboração baseada em mensagens permitem o desenvolvimento de aplicações distribuídas fracamente acopladas com um enorme grau de interoperabilidade.

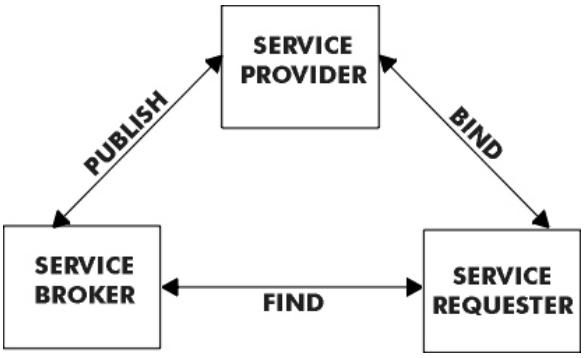

Figura 2 – Arquitetura e integração de Web Services (KREGER, 2001)

Web Services é uma tecnologia que se constitui de outras tecnologias e padrões que estão sendo desenvolvidas há pouco tempo. A Figura 3 mostra estes padrões que especificam meios de troca de mensagens entre serviços (SOAP), busca de serviços em registros (UDDI) e configuração dinâmica de clientes baseadas em descrições de serviços (WSDL). Retrata a interoperabilidade e padrões na qual são desenvolvidos Web Services.

Figura 3 – Pilha de protocolos que compõe os Web Services (KREGER, 2001)

2.7.1 SOAP

SOAP (*Simple Object Access Protocol*) originou-se da idéia de um mecanismo de RPC baseado em XML originalmente proposto por Dave Winer em 1998. O SOAP foi, então, desenvolvido e descrito pela IBM, Lotus Development Corporation, Microsoft, Development-mentor e Userland software e é suportado pela Sun Microsytems.

SOAP é um protocolo baseado em HTTP-XML que permite que aplicativos se comuniquem facilmente pela Internet, utilizando documentos XML chamados Mensagens

SOAP. É compatível com qualquer modelo de objeto, já que incluem funções e capacidades que são absolutamente necessárias para definir uma estrutura de comunicação. Portanto, SOAP é independente tanto de plataforma como de software e pode ser implementado em qualquer linguagem. Suporta transporte utilizando quase todos os protocolos concebíveis. Por exemplo, o SOAP pode ser associado ao HTTP e seguir o modelo de solicitação-resposta de HTTP. O SOAP suporta também qualquer método de codificação de dados (DEITEL, 2003C).

O SOAP pode ser utilizado para fazer uma chamada de procedimento remoto (RPC), que é uma solicitação feita a outra máquina para executar uma tarefa. A RPC utiliza um vocabulário XML para especificar o método a ser invocado, quaisquer parâmetros que o método receba e o URI do objeto-alvo. Uma chamada RPC naturalmente mapeia uma solicitação HTTP, de forma que a mensagem é enviada por meio de um POST HTTP. A mensagem de resposta SOAP é um documento de resposta HTTP que contém os resultados da chamada do método (por exemplo, valores retornados, mensagens de erro e etc). O SOAP também suporta RPC assíncrona, na qual os dados seguem regras diferentes de codificação e não mapeia os parâmetros de uma RPC específica.

No entanto, utilizar RPC na Internet também representa um problema de segurança. *Firewalls* e *Proxys* normalmente irão bloquear este tipo de fluxo na rede. Uma melhor forma de comunicação entre aplicações seria a utilização de HTTP, uma vez que este protocolo é suportado por todos os servidores Web, navegadors e firewalls.

A sintaxe de uma mensagem SOAP é simples. Esta deve ser codificada no padrão XML e contém as seguintes partes:

SOAP Envelope: define o conteúdo da mensagem e os vários namespaces que são usados pelo resto da mensagem, incluindo xmlns:SOAP-ENV (SOAP Evelope), xmlns:xsi (*Xml Schema for Instances*) e xmlns:xsd (*Xml Schema for Datatypes*).

SOAP Header: é um elemento opcional e fornece meios para o processamento adicional em trânsito do remetente para o destinatário. Ele também pode incorporar informações de roteamento. Pelo cabeçalho, protocolos mais complexos podem ser construídos sobre o SOAP. A mensagem pode ser estendida de forma modular por entradas de cabeçalho para propósitos como autenticação, gerenciamento de transação e pagamento. O corpo de uma mensagem SOAP contém dados específicos de aplicativo para o destinatário final da mensagem.

SOAP Body: contém informação a respeito de métodos e parâmetros a serem chamados ou respostas enviadas. Quando SOAP é utilizado como RPC, esta parte possui um único elemento que contém o nome do método, parâmetros e a identificação do serviço.

Mensagens SOAP podem ser enviadas utilizando outros protocolos de transporte, como por exemplo, SMTP.

Uma mensagem SOAP/HTTP é geralmente processada por uma aplicação que está em um servidor Web. Quando a aplicação recebe uma requisição, esta primeiro confere a existência do campo SOAPAction na requisição. Se possuir tal campo, a requisição é direcionada para a SOAP Engine, responsável por processar a parte XML da mensagem e usar o endereço do serviço especificado para executar uma varredura em seu registro local (este registro contém informações a respeito dos serviços disponíveis no servidor). Então, é utilizado reflexão para localizar e invocar o método especificado no serviço e retornar a resposta.

A Figura 4 contém um exemplo de uma requisição SOAP/HTTP utilizada como RPC:

```
POST /soap/servlet/rpcrouter HTTP/1.0
Host: localhost:8070
Content-Type: text/xml
Content-Length: 461
SOAPaction: ""
<SOAP-ENV: Envelope
xmlns:SOAP-
ENV="http:/schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http:/www.w3c.org/1999/XMLSchema-
instance"
xmlns:xsd="http:/www.w3c.org/1999/XMLSchema">
<SOAP-ENV:Body>
<nsl:getRate xmlns:nsl="urn:xmethods-exchange"
SOAP-ENV:
encodingStyle=http://schemas.xmlsoap.org/soap/encodi
ng/>
<country xsi:type="xsd:string">USA</country1>
<country xsi:type="xsd:string">Japan</country2>
</nsl:getRate>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Figura 4 – Formato de uma Requisição SOAP/http (W3C-XML, 2004)

Uma resposta SOAP é retornada em um documento XML estruturado como em uma requisição, exceto quando contém o resultado do método requisitado.

A segurança no envio de mensagens SOAP é um tópico importante. Em um nível mais baixo, mensagens SOAP podem ser trocadas pela rede utilizando HTTPS ao invés de HTTP. Como HTTPS utiliza SSL no seu transporte, fica garantida a proteção contra possíveis intervenções.

Além disso, o cliente e servidor podem verificar cada um suas respectivas identidades. Embora HTTPS resolva o problema de proteção das mensagens contra possíveis invasores, este não ajuda muito quando se necessita da segurança necessária para a autenticação de usuários de Web Services específicos. Estes serviços irão fornecer algum tipo de combinação de usuário/senha durante a fase inicial de registro no serviço e então esta será utilizada para acessos futuros. Os serviços UDDI são um tipo de Web Services que requerem um registro inicial antes da utilização de seus serviços de publicação. Não há ainda um padrão

de autenticação para Web Services, mas tal padrão deverá surgir em pouco tempo devido trabalho de organizações como Microsoft e Verisign.

O apoio industrial ao protocolo SOAP está aumentando a cada dia com o lançamento de produtos que suportam este protocolo pelas maiores organizações. Enquanto cada um destes produtos possuem suas próprias capacidades, todos são capazes de produzir e processar mensagens SOAP. Isto significa que não importa como a aplicação foi implementada, ou onde está localizada, ou ainda em qual linguagem foi implementada, o que interessa é que a interoperabilidade entre aplicações está garantida.

2.7.2 WSDL

WSDL (*Web Services Description Language*) é uma linguagem baseada em XML projetada para descrever Web Services, como o que um serviço pode fazer, onde está localizado e como invocá-lo. Uma descrição de serviço contém uma definição abstrata para um conjunto de operações e mensagens, um protocolo de integração para estas operações e uma localização específica na rede (FERGUSON, 2004).

Sendo baseado em XML, um documento WSDL é dividido em seções constituídas por elementos. O elemento <definition> contém informações a respeito de um ou mais serviços.

Dentro deste elemento, encontram-se outros quatro elementos:

<message> e <portType> : define quais operações o serviço fornece.

<type> : este elemento opcional serve para definir tipos de dados estruturados.

<binding> : designa como as operações são invocadas.

<service> : explicita onde o serviço está localizado.

Um documento WSDL pode ser dividido em documentos distintos, um que se refere à implementação de um serviço e outro que se refere à interface do serviço. Neste caso, o

documento de implementação conterá um elemento <service> e um elemento <import> que apontará para o documento WSDL com a definição da interface do serviço. No documento que define a interface serão encontrados os elementos <types>, <message>, <portType> e <binding>.

2.7.3 UDDI

UDDI (*Universal Description Discovery and Integration*) é uma iniciativa da indústria que permite às organizações publicar e buscar informações a respeito de Web Services. É composto por um grupo de registros baseados na Web que fornecem informações a respeito de uma organização ou entidade. Estes registros são executados em múltiplos servidores (operadores) e podem ser usados por qualquer um que queira tornar disponível informações a respeito de negócios ou entidades, ou ainda por qualquer um que queira buscar estas informações (FERGUSON, 2004).

A informação definida em uma descrição de serviço encontrada em um documento WSDL complementar à informação encontrada em um registro UDDI. Este fornece suporte para vários tipos de descrições de serviços, ou seja, UDDI não possui suporte direto para WSDL ou outro mecanismo de descrição de serviços.

Os quatro tipos de dados encontrados em um registro UDDI são:

businessService: fornece informações sobre uma organização e pode conter um ou mais businessService.

businessEntity: fornece informações técnicas e descrições de serviço . Pode conter um ou mais bindingTemplate.

bindingTemplate: contém uma referência a um ou mais tModels.

tModel: utilizado para definições de especificações técnicas de serviço.

UDDI é uma especificação em constante desenvolvimento. É coordenada por UDDI.ORG, que é composta por vários membros da indústria, como Microsoft, IBM e Ariba. Esta especificação fornece uma API para consulta e publicação de serviços em registros UDDI (FERGUSON, 2004).

A consulta e publicação em registros UDDI é executada utilizando-se mensagens no formato SOAP. Estas operações são baseadas na especificação de uma API proposta por UDDI.ORG e possui mensagens especificas para a busca, publicação e alteração de registros.

2.8 RDF

O surgimento da Web e a relativa facilidade de se criar documentos tem levado a uma abundância de informações veiculadas. Com isso, encontrar dados sobre um tópico específico pode ser difícil e consome tempo. A RDF (*Resource Definition Framework*) é uma linguagem baseada em XML que descreve as informações contidas em um recurso. Pode ser uma página, um site inteiro ou qualquer outro item na Web que contenha informações sob qualquer forma. A “informação da informação” da RDF (ou metadados) pode ser utilizada pelos mecanismos de busca ou pelos agentes inteligentes de software para listar ou catalogar as informações na Web. A RDF pode também ser utilizada para avaliar um site ou para criar assinaturas digitais (isto é, o equivalente digital de uma assinatura escrita). O *Resource Definition Framework Model and Syntax* é uma recomendação do W3C. A *RDF Schema Specification* Version 1.0 é, atualmente, uma Recomendação candidata do W3C (DEITEL, 2003C).

2.9 Web Semântica e Ontologias

A Web Semântica foi criada pelo mesmo cientista que inventou a *WWW - World Wide Web*, o físico inglês Tim Berners-Lee. A Web Semântica estrutura o conteúdo significativo das páginas Web, criando um ambiente em que agentes de software percorrem página por página para executarem tarefas solicitadas pelos usuários. O desenvolvimento da Web Semântica envolve duas importantes tecnologias: XML e RDF (BERNERS-LEE, 2004).

O objetivo é fazer com que os computadores entendam o conteúdo da Web. O primeiro passo será organizar e estruturar as informações e o segundo será adicionar semântica às informações da Web, de forma tal que agentes de software possam compreendê-las.

A Web Semântica constitui-se em uma das formas mais diretas de representação de Ontologias. A Web Semântica é um ambiente informativo, em que os usuários dizem qual o significado da informação, ao invés de dizer o que fazer com ele. Neste contexto, a Web Semântica herda da Web atual a filosofia de navegação, interconexão, distribuição e descentralização. É um repositório de recursos marcados por suas URIs (*Universal Resource Identifier*). Na Web Semântica, a palavra “semântica” significa “máquina capaz de processar significado”, em contraste com a WEB atual, onde o conteúdo apenas faz sentido para o entendimento humano (TORRES, 2003).

A Web Semântica representa o desejo em desenvolver métodos e linguagens para o entendimento do computador sobre o significado dos dados armazenados nele, tornado possível à análise e o processamento destes dados, obtendo conclusões e estabelecendo novas informações. Em outras palavras, a Web Semântica resultará em um grupo de tecnologias que irão habilitar as máquinas a entender as informações na Web, tornando a rede mais fácil de ser consultada e mais amigável com seus usuários (TORRES, 2003).

Para resolver este problema, existem as ontologias que vão fornecer o vocabulário necessário, para a comunicação entre os agentes de software e as páginas da Web e mostrar as relações entre os conceitos.

Portanto, uma ontologia é um documento ou um arquivo em que estão definidas formalmente as relações entre conceitos. Uma ontologia na Web é uma taxonomia formada de classes e subclasses de objetos, relacionadas entre si mais um conjunto de regras de inferência (HENDLER, 2001).

O uso de ontologias permitirá a melhora da velocidade das pesquisas, uma vez que os agentes irão pesquisar somente as páginas que se referem à informação desejada, ao invés de todas as páginas usando palavras-chave ambíguas, conforme realizado atualmente pelas ferramentas de buscas já conhecidas.

O problema maior será criar padrões que sejam usados globalmente. A curto prazo, existirão DTDs padrões para certos tipos de aplicações como por exemplo: e-commerce, sistemas de informação hospitalar, sistemas de bibliotecas entre outras, que possibilitarão aos projetistas/analistas e desenvolvedores de software, a utilização de um padrão denominado “ontologias” para a criação e troca de dados.

Usando RDF não é possível criar ontologias, no entanto, ele é a base para diversas linguagens com essa finalidade, tais como RDFS (*RDF Schema*) (W3C-RDF, 2004), um recurso que define as primitivas para a criação de ontologias.

A especificação do RDF Schema fornece os mecanismos necessários à definição de: elementos, classes de recursos, possíveis restrições de classes e relacionamentos, e detecção de violação de restrições, constituindo uma hierarquia com múltiplas relações de herança.

A especificação do RDF Schema fornece também recursos suficientes para criar modelos RDF, através de sistemas hierárquicos representados por taxonomia de assuntos como mapas de sites, dicionários e esquemas de classificação de bibliotecas.

A Web Semântica sugere que sejam disponibilizadas informações na Web com características descritivas (metadados) que expressem um entendimento semântico de cada fonte de dados nela encontrada (HENDLER, 2001).

Porém, a falta de padronização para definição dos metadados torna sua utilização, conseqüentemente, num sério problema para que se tenha um eficiente intercâmbio de informações na Web.

Muitos desses problemas de definição podem ser resolvidos com a utilização de um vocabulário comum, englobando conjuntos de termos com suas definições. Como proposta para possibilitar um processamento, compartilhamento e reutilização dessas informações semânticas na Web, surgem as Ontologias (DECKER, 2000).

O conceito de ontologias foi inicialmente utilizado na área de Inteligência Artificial (IA) para descrever conceitos e relacionamentos utilizados (conhecimento compartilhado). Uma das definições mais simples e bastante usada de ontologias aplicada ao contexto da Ciência da Computação é a de Tom Grubber, pesquisador em representação e compartilhamento de conhecimento. Para Grubber, uma ontologia é uma especificação explícita de uma conceitualização (GRUBBER, 1993).

Definidas como uma conceitualização formal e comum de um particular domínio de conhecimento, as ontologias fornecem um entendimento semântico comum de tópicos que podem ser utilizados tanto na comunicação entre pessoas como entre aplicações de sistemas (DECKER, 2000). Com as ontologias, será possível especificar domínios de conhecimento, através da definição de conceitos e suas características. Essa fonte de informação, representada pelas ontologias, deverá ser cuidadosamente definida e, evidentemente, surge a necessidade da intervenção humana para alimentá-la, uma vez que ela representará uma interpretação semântica unificada de um domínio de conhecimento (MELLO, 2000).

Com o uso de ontologias, é possível representar informações que refletem um entendimento semântico de diversas situações do mundo real. Entretanto, é preciso considerar que é uma tarefa bastante complicada representar tais situações com toda sua riqueza de detalhes. Para simplificar a definição de ontologias, tem sido utilizado o conceito de representação de domínios, em que é representada uma parte do mundo, restringindo, assim, a diversidade de informações e, ao mesmo tempo, concentrando a definição em cada domínio de conhecimento para permitir uma representação cada vez mais rica do mesmo.

As ontologias se propõem a capturar domínios de conhecimento de forma genérica, para fornecer um entendimento semântico desses domínios que poderá ser utilizado e compartilhado por diversas comunidades e aplicações (STAAB, 2000). Logo, é importante que as ontologias sejam definidas preferencialmente por especialistas de um dado domínio, visto que não há ninguém mais indicado para definir os conceitos-chave de uma determinada área de conhecimento do que pesquisadores desta área.

Uma das principais motivações em se definir ontologias é a de que com isso pode-se obter uma base de informações que poderá ser compartilhada e reutilizada por diversas aplicações (GUARINO, 1997).

Ao definirmos ontologias pode-se definir acordos entre usuários de informação e os fornecedores de informação para padronização da utilização dos dados na Web, diminuindo os problemas de consistência no intercâmbio de dados nesse ambiente.

Esses acordos são chamados Compromissos Ontológicos (*Ontological Commitments*) e permitem que somente determinados significados de um domínio sejam capturados numa dada especificação. O intercâmbio de dados é padronizado e será baseado no modelo de especificação que foi criado.

Sendo assim, ao definir o compartilhamento da utilização de uma dada ontologia, usuários e aplicações associarão regras comuns de padronização aos dados e,

conseqüentemente, assegurará maior confiabilidade às informações disponibilizadas e recebidas.

Os compromissos ontológicos podem definir diferentes graus de detalhe e isso dependerá dos propósitos de cada ontologia, ou seja, a delimitação do escopo da representação de cada domínio de conhecimento dependerá dos propósitos definidos pelos usuários para especificar essa ontologia.

Quando se fala em definir ontologias para representar domínios de conhecimento, considera-se a possibilidade de termos várias ontologias representando diversas fontes de conhecimento, definidas, utilizadas e integradas de acordo com a necessidade da situação. Como não há um consenso que defina um escopo ao qual cada ontologia deverá atender, pode-se pensar na definição de uma ontologia universal, proporcionando um vocabulário global. Entretanto, utilizando uma ontologia universal, seria mais complicado garantir um entendimento semântico unificado numa proporção tão ampla. É evidente que vários problemas decorreriam desde a definição desse vocabulário global, que deveria ser consensual para todos, até a imposição de sua utilização e conseqüentemente sua manutenção.

Portanto, o uso de ontologias específicas para representar domínios de conhecimento é mais recomendado, dada a possibilidade de se estabelecer um consenso, entre os diversos membros que compõem a comunidade desta área, ser mais factível.

Em termos práticos, uma ontologia consiste basicamente na definição de alguns itens:

- Conceitos que representam tópicos importantes na definição de um dado domínio de conhecimento;
- Definição de algumas características relevantes para esses tópicos; e
- Definição de relacionamentos entre esses conceitos, além de possibilitar uma organização hierárquica desses conceitos, sempre que for necessário.

As ontologias podem ser classificadas da seguinte forma (GUARINO, 1997):

- Ontologias de nível superior – descrevem conceitos gerais como espaço, tempo, objeto, assunto, ação e/ou metadados como entidades, relações e atributos;
- Ontologias de domínio e de tarefa – podem ser especializações da ontologia de nível superior, descrevendo, respectivamente, um vocabulário para um universo de discurso (medicina ou automóveis) e para uma tarefa genérica (diagnóstico ou venda);
- Ontologias de aplicação – essas dependem tanto de um domínio quanto de uma tarefa particular, podendo ser uma especialização de ambas. Corresponde a regras impostas por conceitos de domínio quando executam uma certa tarefa, como por exemplo, substituição de uma unidade sobressalente de um automóvel.

As ontologias podem ser utilizadas em vários serviços na Web. Um exemplo da sua utilização é em sites de comércio eletrônico (*e-commerce*).

As ontologias são utilizadas para fornecer uma base de informações comum e padronizada, englobando os conceitos-chave utilizados pelos serviços requisitados por cada contexto. No caso do comércio eletrônico, essas informações são utilizadas para integrar e unificar as definições de produtos oferecidos por várias lojas de forma padronizada, além de reutilizar diferentes descrições para um mesmo produto oferecido por diferentes lojas virtuais (isso é possível, pois, uma vez que as ontologias realizam relacionamentos entre conceitos, pode-se dispor de conceitos descritos de forma diferente sintaticamente, mas similares semanticamente).

Assim vários serviços de compra e venda podem ser disponibilizados de forma completamente automatizada. Logo que os produtos oferecidos pelas lojas estejam descritos sob uma forma semanticamente comum entre elas, tornar-se-á possível uma comunicação de cliente vendedor entre máquinas diretamente, sem a direta interferência humana.

Uma outra aplicação, em que as ontologias vêm sendo bastante utilizadas, está ligada a área de banco de dados e recuperação de informação, como suporte à interoperabilidade de

fontes de dados distribuídas e heterogêneas (ARRUDA, 2002). As ontologias têm sido aplicadas principalmente em banco de dados heterogêneos e *Data Warehouses* como modelos conceituais globais, resultantes de uma concordância na definição de entidades e relacionamentos.

As ontologias podem ser utilizadas nas máquinas de busca (*search engines*) da Web servindo como um esquema conceitual que dá suporte semântico às consultas (MELLO, 2000). Ao utilizar ontologias, as máquinas de busca podem fazer suas consultas através de suas palavras-chave e como resposta a essa consulta pode-se ter, além das páginas que contêm a palavra-chave informada, outras páginas contendo informação sintaticamente diferente da palavra-chave, ou seja, sinônimos que são escritos de outra maneira, mas que expressam o mesmo significado semântico (DECKER, 2000). Por exemplo, numa consulta com palavra-chave “carro” poderia retornar páginas que contenham a palavra “automóvel”.

As ontologias também podem ser utilizadas para guiar processos de extração de dados semi-estruturados através da manutenção de informações que auxiliem na identificação de atributos de conceitos e relações. Como no gerenciamento de dados semi-estruturados, a extração de dados faz parte do processamento de uma consulta, algumas aplicações, que provêem essa facilidade, utilizam ontologias como esquemas conceituais a partir do qual consultas com caráter semântico podem ser formuladas (MELLO, 2000).

2.10 Ferramenta NetBeans IDE

NetBeans é um projeto open-source, consistindo em um IDE profissional cheio de recursos e uma plataforma sobre a qual você pode desenvolver qualquer tipo de aplicação. O NetBeans foi adquirido e colocado como open-source pela *Sun Microsystems*, principal incentivadora do projeto e que o usa como base para o Sun ONE Studio (NETBEANS, 2003).

NetBeans IDE é um ambiente de desenvolvimento profissional escrito em Java. Ele pode ser usado para desenvolver código em Java, HTML, XML, JSP, C, C++ e outras linguagens.

O IDE é modular, e existe uma enorme variedade de extensões gratuitas e comerciais para ele adicionando suporte a diversas tecnologias. Podem ser citadas as seguintes características:

- Editor de código com destaque de sintaxe avançado;
- Suporte a tecnologias JSP, XML, RMI, CORBA, JINI, JDBC e Servlet;
- Suporte a Ant, CVS e outros sistemas de controle de Versão;
- Suporte conectável para compiladores, depuradores e serviços de execução;
- Assistentes e ferramentas de geração de código e gerenciamento.

O IDE NetBeans foi utilizado no desenvolvimento deste projeto devido as características citadas, além de integrar em um único ambiente suporte para as linguagens Java, XML e mecanismos para teste e implementação com banco de dados.

Além do IDE NetBeans, para garantir poder de processamento às solicitações do lado cliente, optou-se pelo uso de sistemas de *clusters* baseado na arquitetura Beowulf.

CAPÍTULO 3 – COMPUTAÇÃO PARALELA E DISTRIBUÍDA

3.1 Introdução

Um dos principais objetivos da proposta do SISPRODIMEX é a redução no tempo de resposta às solicitações do cliente. Em se tratando de distribuição de informações e imagens médicas, pode-se supor que sempre existe a necessidade de alto poder de processamento.

Considerando todas as limitações que envolvem a transmissão de dados em uma rede local ou mesmo usando os recursos da Internet, a Lei de Amdahl sugere uma possível solução inicial para este problema associada a outras soluções que surgem através da utilização da linguagem de programação Java e do uso de *clusters* de computadores ou até mesmo de sistemas *Grid-computing*.

Apesar do quanto promissor a computação paralela possa parecer, ela não é uma solução para todo o problema de processamento. Existem tarefas que são eminentemente seqüenciais e que não tiram proveito de um computador paralelo. É comum que as tarefas a serem executadas possuam porções paralelizáveis e porções que precisam ser executadas de forma seqüencial. É importante observar que um computador paralelo operando de forma seqüencial é um grande desperdício, pois, enquanto um processador trabalha no trecho serial, todos os demais ficam ociosos.

3.2 A Lei de Amdahl

Abordando esse tema, Amdahl propôs uma expressão para esse problema, que ficou conhecida como Lei de Amdahl (AMDAHL, 1967) (PACHECO, 1997) e que está representada a seguir. A explicação é muito simples. Imagine uma tarefa que executada em uma máquina seqüencial gaste "T" segundos. Porém, quando preparada para execução em uma máquina paralela, ela tem uma porção que obrigatoriamente deve ser executada de forma serial. Supondo que "p" represente a porção serial, em conseqüência "(1-p)" representa a porção paralelizável. Quando executado em uma máquina paralela com N processadores, o tempo gasto será igual a

$$pT + (1-p)\frac{T}{N}$$

Somente o trecho paralelizável tira partido dos "N" processadores. Assim o Ganho de Processamento é dado por:

$$GP = \frac{T}{pT + \frac{(1-p)T}{N}}$$

ou

$$GP = \frac{1}{p + \frac{(1-p)}{N}}$$

que é a forma consagrada da Lei de Amdahl. Deve-se notar que neste caso esta sendo ignorado os custos de sincronização e de troca de parâmetros entre os processadores. Pode-se observar em um gráfico o desempenho de alguns programas, segundo essa lei. A Figura 5 apresenta 2 casos. As barras claras apresentam uma tarefa 100% paralelizável, ou seja, p = 0. Nesse caso o aumento do número de processadores reflete linearmente no desempenho. As barras escuras representam o caso em que 1% (p = 0,01) da tarefa precisa ser executada de

forma serial. Nota-se uma grande queda de desempenho para grande quantidade de processadores, ocorre que há uma saturação, e que o aumento de processadores, por exemplo, de 50 para 100 processadores, pouca diferença traz para o desempenho.

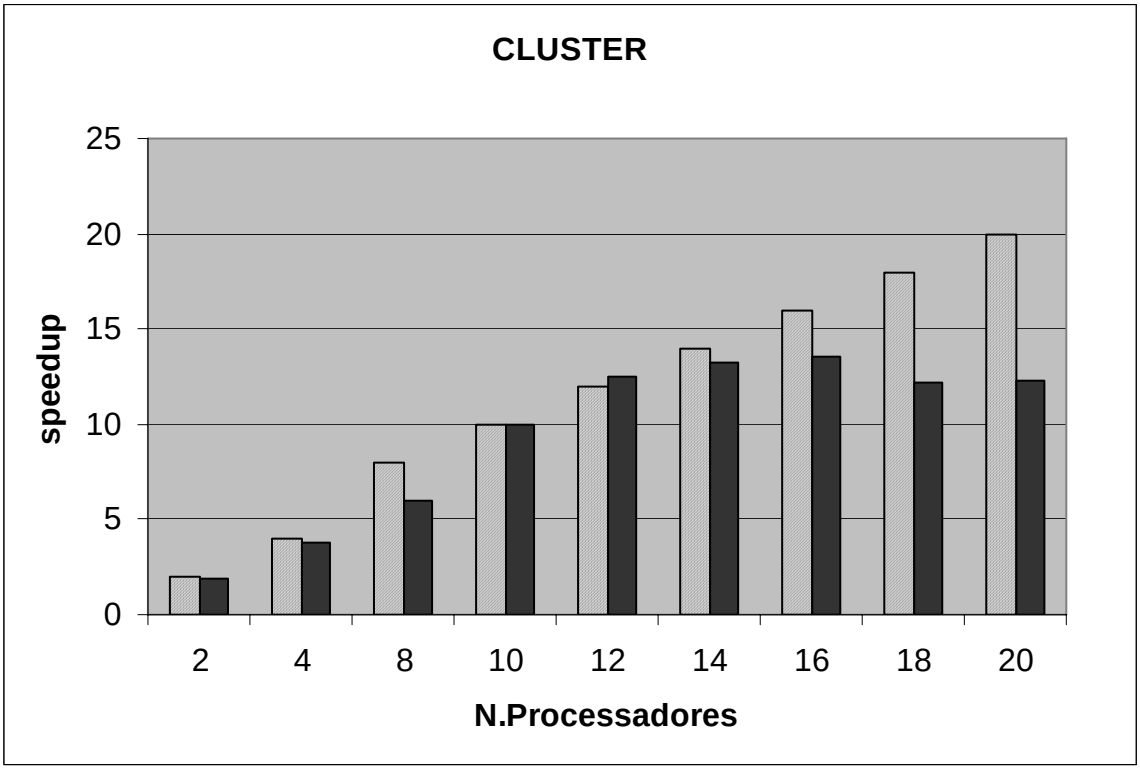

Figura 5 – Ganho de performance de acordo com o número de processadores (MUCHERONI, 2004)

Devem ser ressaltados dois pontos importantes. O primeiro é que, exceto para o caso de p = 0, todos os demais casos irão apresentar saturação em algum ponto. Se p é pequeno, a saturação só ocorre com um elevado número de processadores. O segundo ponto é que, contrastando com a Figura 5, a Lei de Amdahl não prevê queda no desempenho, no pior caso, ele fica fixo em algum valor.

3.3 Computação Distribuída Baseada em Java

Entre os novos conceitos expostos neste trabalho a respeito da linguagem Java, pode-se enfatizar o uso de *threads* e suporte a sockets (SUN, 2001) (SUN, 2002).

Pesquisadores da Universidade de Brasília (UNB) efetuaram experimentos comparativos para demonstrar o comportamento de uma aplicação Java em um cluster baseado na arquitetura Beowulf e em ambientes heterogêneos (SOUZA, 2003). Para tal experimento foi proposta a multiplicação de duas matrizes de inteiros (A e B) de 2048 X 2048 elementos. A comunicação entre os nodos processadores foi baseada na construção de sockets do Java.

Em um outro projeto de mestrado, correlato ao SISPRODIMEX, a mesma implementação de um cluster baseado na arquitetura Beowulf foi efetuada obtendo-se praticamente os mesmos resultados apresentados. Tal cluster está sendo utilizado no momento na tentativa de se obter resultados para processamento de objetos DICOM em ambientes de *clustering* com Java-MPI (PRIMO, 2005).

A arquitetura Beowulf é uma categoria especial de cluster contruídas a partir de máquinas e dispositivos de baixo custo facilmente encontrados no mercado. A idéia principal é que seja feita uma configuração de hardware e software para que se tenha um aumento de performance na execução paralela de aplicações. A categoria Beowulf está dentro do grupo dos sistemas SSI (*Single System Image*), ou seja, o usuário deve ver o sistema como uma máquina única (WALKER, 2001).

Uma característica importante que se tem em máquinas Beowulf é a centralização dos pacotes de programas e contas de usuários em uma máquina master que é a responsável pelas tarefas administrativas do cluster. Quando uma tarefa é enviada ao cluster, a máquina master recebe e divide entre os nodos processadores, também chamados de escravos. Sendo

assim, um cluster Beowulf de 16 nodos, na verdade, possui um total de 17 máquinas sendo 1 master e 16 escravos (SOUZA, 2003).

O teste com a arquitetura Beowulf aqui referenciado foi feito com uma máquina com 8 nodos processadores Pentium-III 350, 196Mbytes de memória RAM, HDs de 6Gbytes e comunicação entre os nodos baseada em placas de rede padrão Ethernet 10/100mbps (SOUZA, 2003).

Os dados obtidos e apresentados na Tabela 1 mostram um ganho considerável de tempos de execução a medida em que se aumenta o número de processadores.

Tabela 1 – Tempos de execução para 1, 2, 4 e 8 processadores (SOUZA, 2003).

PROCESSADORES			
1	**2**	**4**	**8**
1225,99 (s)	634,95 (s)	333,48 (s)	190,56 (s)

Já na Figura 6, percebe-se que apesar do ganho considerável de tempos de processamento, com o aumento do número de processadores ocorre uma pequena perda de speedup quando há o aumento de 4 para 8 processadores. Tal resultado vem a confirmar as teorias expostas na Lei de Amdahl.

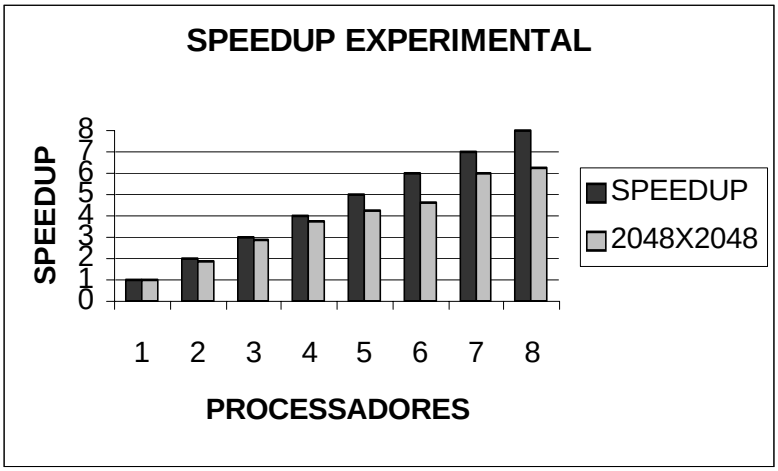

Figura 6 – Speedup medido (SOUZA, 2003)

O cluster está ligado à rede local do laboratório da universidade e este apresenta estações de trabalho de outras plataformas como IBM RS6000 e Powerpc, por exemplo. Para continuidade dos experimentos foram considerados os ambientes Windows NT e Linux.

Primeiramente foi feita a multiplicação de matriz usando um único processador no Linux e depois no Windows NT. O objetivo foi ver os tempos de execução nas duas plataformas. O processo sempre foi disparado pelo processador master do cluster Beowulf.

Os tempos de execução foram praticamente iguais (1224,52s no Linux e 1233,60s no Windows NT). O próximo passo foi ver a execução usando um número de 8 processadores, sendo 5 Linux e 3 Windows NT, procurando assim, caracterizar um ambiente heterogêno.

De acordo com os resultados obtidos, na Figura 7 pode-se perceber que o cluster Beowulf obteve uma performance melhor que o ambiente heterogêno.

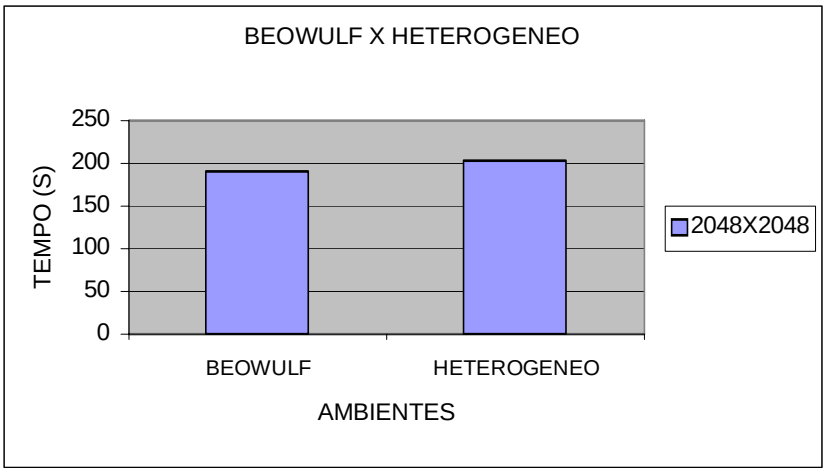

Figura 7 – Execução Beowulf e Ambientes Heterogênos

E, finalmente, foi feita a comparação de execução da aplicação Java com a mesma tarefa sendo executada no MPI (*Message Passing Interface*). A distribuição MPI usada foi Mpitch 1.2.2 do Laboratório Argonne das Forças Armadas Americanas. O teste foi feito usando um tamanho de matriz de 512 X 512, pois, para valores maiores que esse, o MPI retornou um erro de "Falha de Segmentação (*Core Dumped*)". A Figura 8 mostra os resultados obtidos.

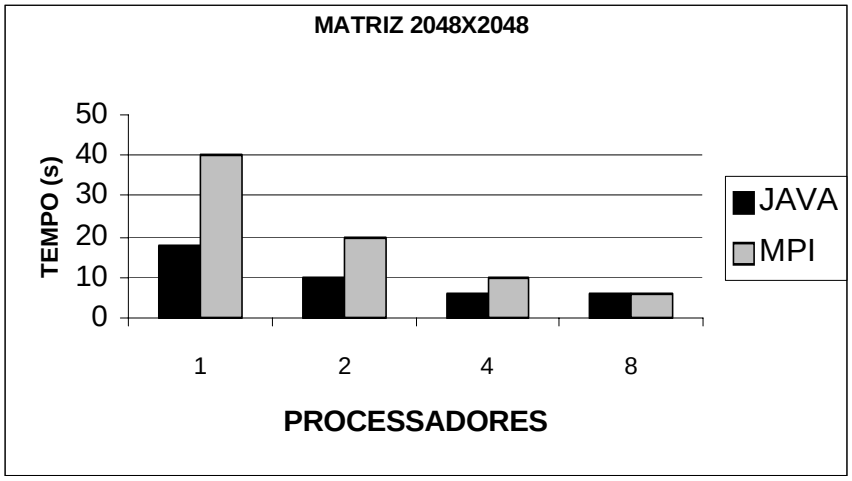

Figura 8 – MPI X Java

Pode-se concluir que a combinação Java com cluster (no caso, a arquitetura Beowulf) mostrou-se uma excelente alternativa para a construção de aplicações distribuídas, em que se obteve um ganho em speedup bem próximo ao linear utilizando 8 nodos processadores (SOUZA, 2003).

CAPÍTULO 4 – SISPRODIMEX – UM SISTEMA DE DISTRIBUIÇÃO DE IMAGENS MÉDICAS

4.1 Introdução

Existe uma enorme atenção voltada para o desenvolvimento tecnológico baseado em imagens médicas como, por exemplo, aquisição, gerenciamento e distribuição. Tal evolução, no entanto, tem gerado um número cada vez maior de informações textuais e, principalmente, de imagens médicas. Conseqüentemente é preciso criar novos meios para gerenciamento destas informações, bem como, minimizar os tempos de respostas aos clientes em relação a estes dados.

Um passo inicial para a criação de um sistema EHR seria a conversão de imagens médicas dos inúmeros sistemas PACS em documentos XML. Tal proposta deixa implícito que várias aplicações possam ser sugeridas (como a administração de agentes de contraste e procedimentos de doses de radiação baseados na análise de dados de imagens médicas) nos sistemas EHR. Este exemplo deixa claro que, sozinho, o DICOM não pode explorar totalmente a riqueza de informações que os dados de imagens médicas fornecem nos ambientes PACS (COHEN, 2004).

E também, considerando que o diagnóstico médico através de imagens digitais tem contribuído grandemente para o avanço tecnológico na área da medicina, pode-se constatar a necessidade de democratização de acesso às informações geradas por sistemas PACS. O grande problema desta democratização é justamente o elevado custo gerado pela implantação de um sistema PACS e pelas dificuldades de acesso a tais informações com independência de localização geográfica.

O SISPRODIMEX (Sistema de Processamento Distribuído de Imagens Médicas com XML) é o nome atribuído à plataforma cuja proposta é disponibilizar, através das modernas tecnologias de comunicação e processamento (de relativo baixo custo), acesso a imagens e informações textuais originadas a partir de objetos no padrão DICOM 3.0, propondo a otimização do impacto desta interação homem-máquina através de implementação de técnicas de Web Semântica e Ontologias baseadas em XML para gerenciamento de contexto.

Acredita-se que a implementação completa da plataforma SISPRODIMEX venha a possibilitar que centros hospitalares de todos os portes (através de médicos e pessoal devidamente credenciado) possam ter acesso a imagens e informações textuais independentemente de localização geográfica através de recursos da Internet e *World Wide Web*.

Uma vez que se vislumbra a criação de uma base fortemente enraizada e organizada sistematicamente para fornecer suporte a pesquisas e tratamentos mais efetivos, as informações oriundas de sistemas PACS são armazenadas em um sistema de *Data Warehouse* que irá, especificamente, armazenar imagens e informações textuais médicas (decodificadas a partir de objetos DICOM) e conta com um módulo de processamento distribuído baseado em *clusters* a fim de minimizar os tempos de respostas às solicitações clientes.

A implementação do SISPRODIMEX, no estágio atual, corresponde a evolução de um sistema de transporte de imagens médicas através do uso de técnicas de serialização e desserialização em linguagem Java, implementados originalmente em applets (CORREIA, 2004). O SISPRODIMEX atual teve sua implementação feita, a partir deste ponto, em linguagem JSP *(Java Server Pages*), através da ferramenta NetBeans IDE.

A implementação completa do SISPRODIMEX prevê o acréscimo de funcionalidades, pois está intimamente relacionado a outros projetos de pesquisa de mestrado, no programa de pós-graduação da UNIVEM, e também de graduação, envolvendo estudos

com arquiteturas de processamento distribuído baseados em *clustering*, *Data Center*, Web Services, Web Semântica e Ontologias.

Portanto, é parte da proposta inicial é demonstrar as ambições do SISPRODIMEX, sua contribuição ao mundo acadêmico, parte inicial de sua implementação e exposição de resultados colhidos. Reconhecer as dificuldades e deficiências das tecnologias propostas para a formação desta plataforma também é um ponto relevante deste trabalho.

4.2 Elementos Motivadores

Para idealizar o SISPRODIMEX foram consideradas as dificuldades iniciais, previstas algumas tecnologias para a construção desta plataforma e as necessidades do mercado.

Foram levadas em consideração as seguintes dificuldades iniciais:

- Gerenciamento da grande quantidade de informações textuais e de imagens médicas geradas em ambientes PACS.
- Dificuldades de obtenção de histórico médico, o que dificulta um diagnóistico mais preciso e obtenção de dados estatísticos.
- O elevado custo inicial de implantação de um sistema PACS e complexidade do padrão DICOM. Os usuários, normalmente, desconhecem a complexidade do padrão DICOM (médicos em sua maioria) e, por sua vez, não querem se preocupar com ele. Ocorre também que nem todos os equipamentos que "falam" DICOM entendem DICOM completamente, portanto, podem ainda, ocorrer problemas de comunicação. Para minimizar este problema existe o *DICOM Conformance Statements* (WHITE-SAND, 2003).

- Necessidade de equipamento de altíssimo desempenho para processamento de imagens.
- Padrão DICOM consome grandes áreas em sistemas de armazenamento e a taxa de compressão de imagens é muito baixa e nem sempre recomendada (no máximo 30%, quando ocorre) a fim de evitar perdas na qualidade de imagem (WHITE-SAND, 2003).
- Dificuldade de acesso à tecnologia DICOM por pequenos hospitais e entidades clínicas em localidades geográficas não privilegiadas.
- Inexistência de um banco de dados orientado a assuntos com a finalidade de facilitar pesquisas dos mais diversos fins.

A arquitetura proposta pelo SISPRODIMEX é baseada em tecnologias abertas, confiáveis, multiplataforma e de relativo baixo custo. Foram levadas em consideração as seguintes tecnologias para a construção da plataforma do SISPRODIMEX, cujos conceitos são desenvolvidos a seguir.

O padrão DICOM 3.0 constitui-se no padrão de armazenamento de imagens médicas e informações relacionadas, e alguns equipamentos de modalidade geram as imagens digitais neste padrão. Mesmo imagens geradas em equipamentos analógicos podem ser convertidas para o padrão DICOM.

O desenvolvimento na linguagem e no ambiente Java foi feito para garantir uso de arquiteturas multiplataforma e implementação com o uso de recursos de programação que representem bem o paradigma orientado à objetos permitindo um código limpo e com boas regras de programação. Parte da dos recursos funcionais são implementados utilizando-se Servlet e JSP

A idéia de compor um banco de dados orientado a assuntos, um *Data Warehouse*, tem como objetivo centralizar as informações clínicas e médicas com claros benefícios de

permitir e facilitar pesquisas a todo o pessoal credenciado, eliminar (ou minimizar) o "gargalo" de armazenamento de objetos DICOM em um ambiente PACS.

Mesmo não sendo o objetivo impor um determinado sistema operacional ou qualquer outro tipo de software (mesmo que isso se faça necessário em determinados momentos, como é o caso do Java), a escolha pelo sistema operacional Linux ocorreu devido ao fato dele ser gratuito, baseado em plataforma Unix e, portanto, estável e confiável, e por oferecer uma gama de ferramentas que colaboram para a criação de uma plataforma segura de trabalho. E essa afirmativa é verdadeira também no que diz respeito à implementação do cluster baseado em Linux.

Partindo da idéia de serviços Web e do uso de Servlets e JSP, a escolha de um servidor de conteúdo recai sobre o Jakarta Tomcat, pois, este servidor estará encarregado de receber e responder as solicitações feitas a partir de um navegador Internet.

Com o uso de arquiteturas baseadas em *clustering* espera-se minimizar o impacto, no lado cliente, de solicitações que exijam alto poder de processamento.

Na camada de dados, os objetos DICOM são decodificados e o acesso a eles e as informações geradas a partir daí são controladas por um sistema gerenciador de banco de dados padrão SQL com a finalidade de agilizar o processo de pesquisa e recuperação de dados.

As respostas disparadas pelo servidor SISPRODIMEX aos clientes serão em formato XML, e por extensão usando Web Services, o que garante a independência da plataforma que estará recebendo estas informações e, caso seja necessário, o lado cliente poderá tratá-la como bem entender.

Como é feita a projeção do uso de um *Data Warehouse* para informações médicas e clínicas baseadas em diagnóstico por imagens, os recursos de Web Semântica e Ontologias terão papel fundamental agilizando o processo de pesquisa e recuperação de dados de uma

forma mais direta e objetiva. A implementação de recursos de gerenciamento contextual se baseia nestas tecnologias.

E, finalmente, foram levadas em consideração as seguintes necessidades do mercado para a construção da plataforma do SISPRODIMEX:

- Necessidade de uma plataforma que viabilizasse o gerenciamento do montante de informações geradas em ambientes PACS.
- Necessidade de democratização de acesso à tecnologia DICOM de objetos de imagens médicas.
- Permitir a criação de uma plataforma com o objetivo de minimizar o impacto de respostas na integração Homem-máquina.

4.3 Uma Visão Geral da Arquitetura do SISPRODIMEX

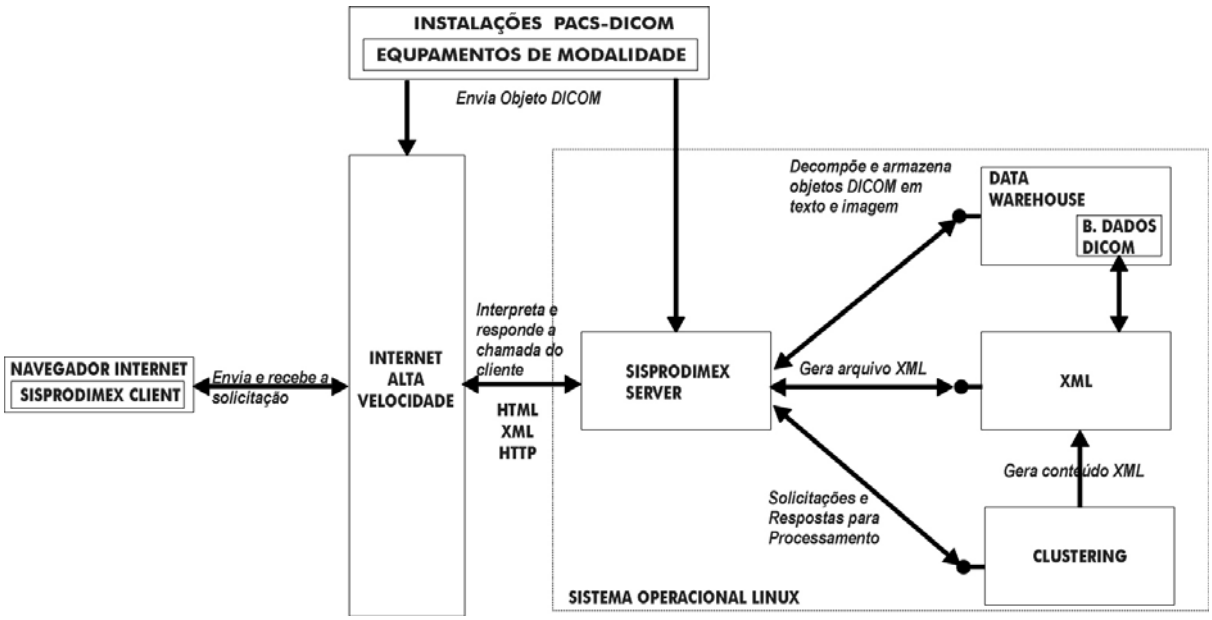

Figura 9 – Modelo da Arquitetura do SISPRODIMEX

Como pode ser observado na Figura 9, a partir de uma solicitação feita através do SISPRODIMEX-Client, por meio de um navegador Internet, o SISPRODIMEX-Server receberá a solicitação, irá processá-la e responder ao cliente da forma mais adequada. Pode-se

observar que o lado Server recebe informações para o ambiente PACS tanto através de uma conexão Internet ou diretamente, via interface de rede local. É enfatizado no modelo o uso de Internet de alta velocidade devido ao grande volume de dados previsto a cada solicitação, correspondendo, inclusive, este item, em um dos pontos críticos que limitam a performance do SISPRODIMEX. Neste modelo, o uso de sistemas de processamento distribuído, como é o caso do pequeno *cluster* implementado baseado na arquitetura Beowulf, tem como objetivo obter ganhos significativos de performance nas respostas aos clientes.

O SISPRODIMEX promove a troca de dados entre cliente e servidor, principalmente, via HTTP. Neste caso, utilizando a tecnologia de Web Services através do SOAP para a transmissão de documentos XML.

Na camada de dados estarão as funções inerentes ao gerenciamento da base de dados. Para facilitar e acelerar o processo de busca por informações na base de dados será utilizado um servidor de banco de dados padrão SQL. Em nossos testes iniciais de implementação utilizou-se o Firebird SQL Server 1.5, por ser um software de banco de dados consistente, multiplataforma, de fácil instalação e manutenção, além de ser gratuito para uso comercial ou não. Na fase de implementação de banco de dados, optou-se por não descrever as regras de negócio utilizando *Stored-Procedure* e *Triggers*. Isso é feito em Java para, assim, manter ao máximo o nível de portabilidade para testes com as demais bases de dados. Não é objetivo do SISPRODIMEX impor uma plataforma de banco de dados.

Na camada de aplicação estarão as funções que irão manter toda a estrutura de funcionamento do sistema de comunicação e as regras de negócio que irão controlar os serviços de pesquisa e recuperação de dados requisitados pelos clientes e todo o controle de acesso ao sistema. Esta camada terá uma característica de *Multithreaded*, pois, é previsto o atendimento de vários usuários e execução de várias tarefas simultaneamente. O

gerenciamento de acesso a dados e composição de respostas ao cliente ficam por conta do servidor Jakarta Tomcat.

Os clientes podem interagir com o SISPRODIMEX através do navegador Internet podendo requisitar os serviços de pesquisa, recuperação, exibição, processamento e, talvez até, a manipulação de informações na base do servidor SISPRODIMEX. Uma vez que o usuário tem a seu dispor informações clínicas de diversos pacientes, o acesso ao servidor SISPRODIMEX estará restrito a usuários devidamente cadastrados com diferentes níveis de acesso, pois, devem ser considerados aspectos éticos, privativos e, mesmo, a segurança das informações.

O uso da linguagem JSP tem como objetivo tornar o SISPRODIMEX um serviço Web para o cliente, de forma que ele não tenha a preocupação de estar acessando uma base de dados DICOM e nem mesmo de estar requisitando um serviço que exija processamento distribuído quando isso for necessário. Evidentemente que o usuário poderá solicitar um objeto DICOM para que ele mesmo possa processá-lo através de um software *DICOM Reader*.

A atual implementação do SISPRODIMEX procura demonstrar as seguintes funcionalidades:

- A partir de objetos DICOM recebidos de um ambiente PACS (com a utilização do software ImageJ) decodificar objetos DICOM em imagens e informações textuais, e criar um sistema de controle de requisições a estas informações a partir de um banco de dados padrão SQL.
- Receber e realizar as solicitações de estudos parciais ou completos de pacientes.
- Efetuar, de acordo com a necessidade, o processamento em um sistema de processamento distribuído baseado em *clusters* Beowulf utilizando JMPI-PLUS.
- Uma vez recebidas às solicitações dos clientes, a aplicação irá processar a solicitação e disparar as respectivas respostas.

4.4. Metodologia de Desenvolvimento do SISPRODIMEX

Após o processo de arquisição de imagens médicas em um ambiente PACS, os objetos DICOM gerados, a partir daí, podem ser recebidos diretamente de uma rede local ou através da Web pelo servidor SISPRODIMEX para, então, ter início o processo de composição do banco de dados de imagens e informações médicas através da decomposição do objeto DICOM.

Em sua implementação completa, o SISPRODIMEX poderá efetuar sozinho o processo de decomposição do objeto DICOM em imagens médicas e informações textuais. Isso, graças ao uso da biblioteca dcm4che, uma implementação do padrão DICOM de autoria da empresa Tiani MedGraph AG (TIANI, 2005).

No entanto, no processo de testes, para simular o comportamento de tal tarefa, utilizou-se a ferramenta de processamento e análise de imagens baseado em Java denominada ImageJ (IMAGEJ, 2004), da autoria de Wayne Rasband do *Research Services Branch, National Institute of Mental Health*, Bethesda, Mariland, USA (RASBAND, 2005).

O ImageJ é um processador de imagem de domínio público baseado em Java e com ele foi possível separar as informações textuais (já em formatos XML e HTML) e as imagens médicas de objetos DICOM (IMAGEJ, 2004).

A Figura 10 mostra a interface da ferramenta ImageJ quando um objeto DICOM gerado a partir de um equipamento de Ressonância Magnética é analisado.

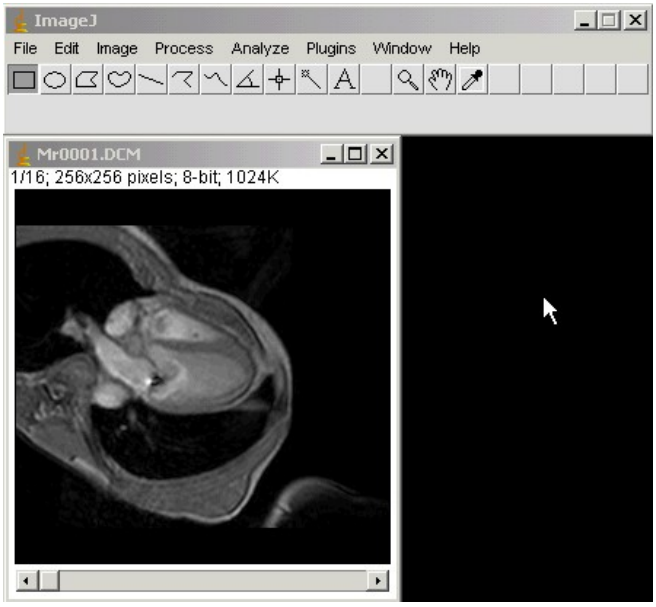

Figura 10 – Objeto DICOM sendo analisado pelo ImageJ

Os arquivos resultantes são, então, copiados para seus respectivos diretórios e mapeados em um sistema gerenciador de banco de dados padrão SQL.

Optou-se pela utilização do Firebird SQL Server 1.5 e para efetuar a conexão com o banco de dados o driver utilizado foi o pacote JayBird JCA/JDBC (FIREBIRD, 2004). Para a manutenção e gerenciamento do banco de dados esta sendo utilizada a ferramenta de distribuição livre para uso não-comercial IBExpert (IBEXPERT, 2005).

Os objetos DICOM utilizados para nos testes realizados foram obtidos por processo de download de uma coleção de imagens disponibilizados gratuitamente no site de Sebastien Barré (BARRE, 2004).

Um usuário poderá fazer acesso ao sistema por meio de um navegador Internet no endereço que irá disponibilizar os serviços do SISPRODIMEX. Ele irá se deparar com uma solicitação de autenticação de usuário. Tal funcionalidade básica também tem como finalidade impor os devidos limites de acesso à base de dados do sistema.

A Figura 11 mostra a interface inicial do SISPRODIMEX e que efetua a autenticação do usuário:

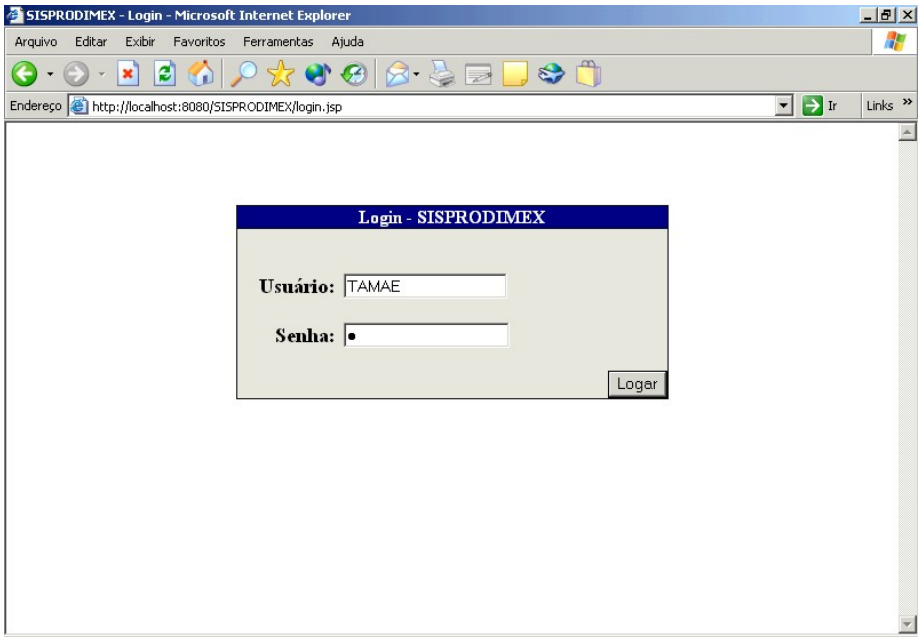

Figura 11 – Tela inicial de autenticação do usuário no SISPRODIMEX

A seguir é, então, disponibilizada a página inicial de serviços disponíveis ao usuário devidamente autenticado no sistema. Como pode ser observado na Figura 12, o usuário poderá efetuar o cadastramento de pessoas autorizadas a terem acesso a base do sistema, efetuar consulta a estudos de pacientes, relatório de estudos de pacientes e, finalmente, uma opção de saída do sistema.

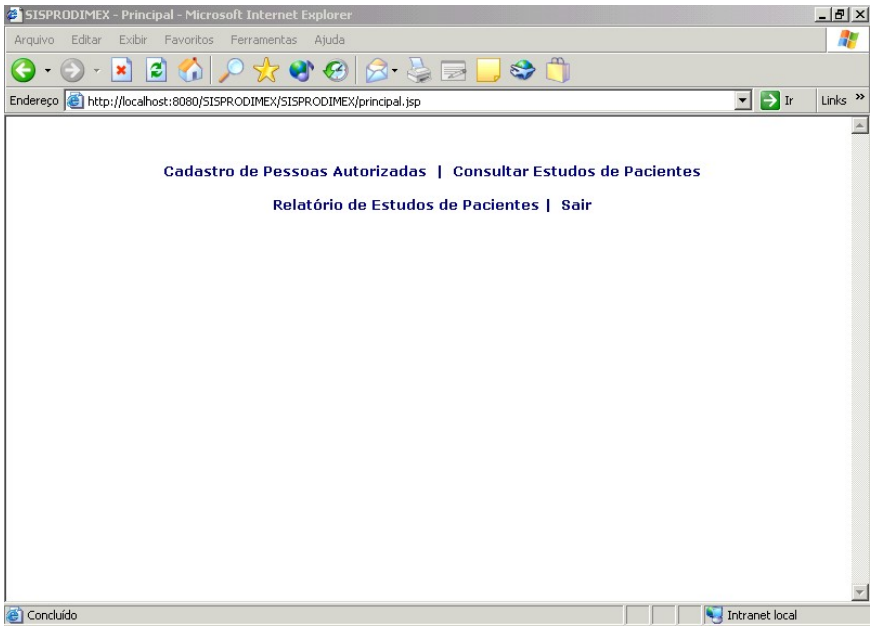

Figura 12 – Serviços disponíveis ao usuário do SISPRODIMEX

A Figura 13 exibe a interface de Consulta de Estudos de Pacientes. O usuário deverá digitar a identificação do paciente desejado para consulta. O sistema fará uma consulta à base de dados para verificar a existência ou não de estudos deste paciente, através de uma declaração SQL-SELECT.

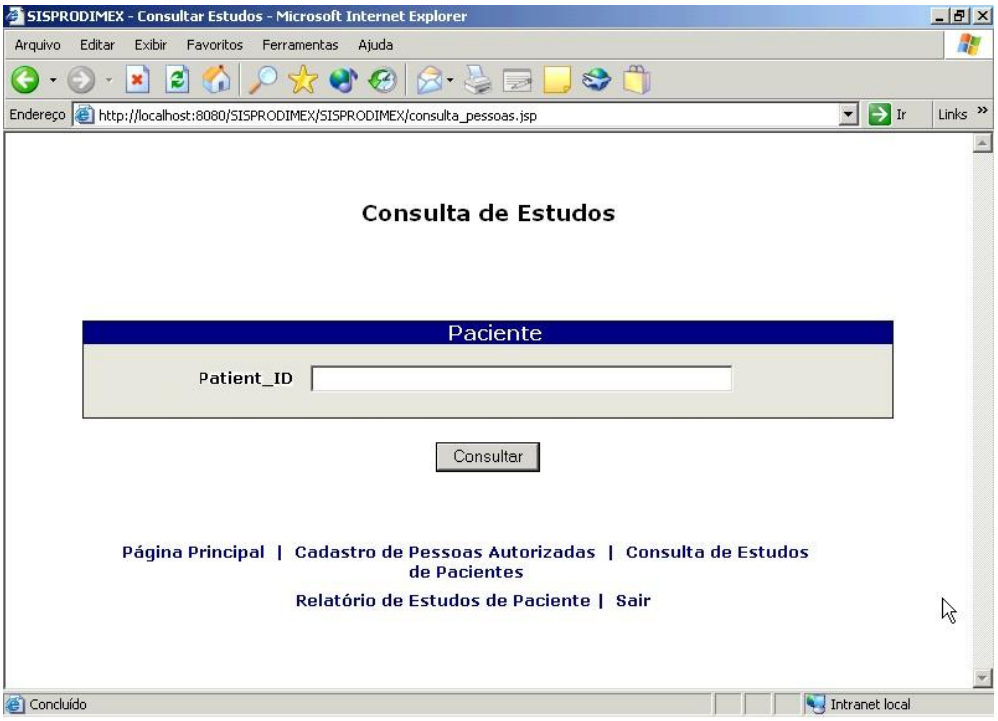

Figura 13 – Consulta de Estudos de Paciente

Considerando a existência do estudo solicitado, de acordo com a interface exibida pela Figura 14, a partir da seleção de um item disponível ao usuário, o SISPRODIMEX enviará um arquivo DICOM (que o processa via JMPI-PLUS no cluster Beowulf), as imagens médicas ou um arquivo com informações textuais em formato XML ou HTML. Pode-se perceber que o nome do paciente sofre um processo de encriptação para proteção dos dados referentes ao paciente.

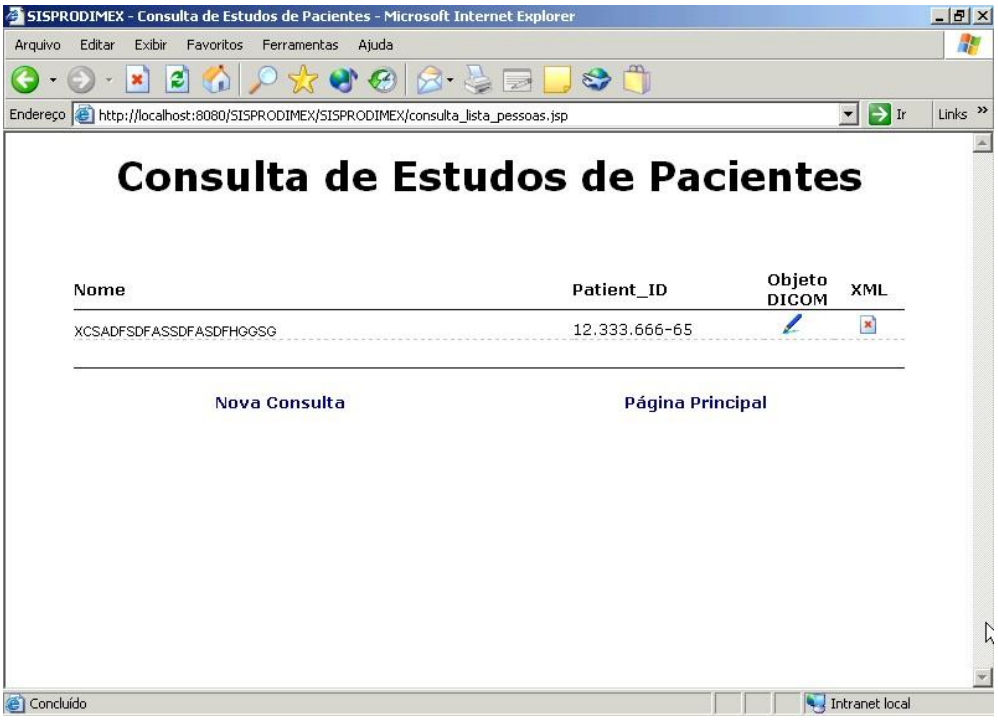

Figura 14 – Resultado da busca por estudo de paciente

4.5 Resultados Iniciais Obtidos

O servidor do SISPRODIMEX foi projetado para ser um servidor *Multithreading*, ou seja, ele possui a capacidade de receber várias conexões de clientes simultaneamente, evitando com isso que um cliente qualquer monopolize o serviço, conectando-se com ele por um longo tempo. Neste ponto, trata-se como cliente o acesso ao objeto DICOM a ser processado pelo cluster. O interessante é que Java é a única linguagem de programação de uso geral e popular que permite especificar atividades simultâneas. O conceito é muito simples, a aplicação específica que os aplicativos contém *fluxos de execução* (*threads*), cada *threads* designando uma parte do programa que pode ser executado simultaneamente com outras *threads*. O servidor do SISPRODIMEX como dá suporte a *multithreads* em uma classe derivada que já é derivada de uma classe diferente de **Thread**, ele implementa a interface **Runnable**, porque em Java não é permitido que uma classe estenda mais de uma classe ao

mesmo tempo. Quando o aplicativo SisprodimexServer é executado, o método **main** cria um objeto **SisprodimexServer**. O construtor configura a interface do usuário que pode ser observada na Figura 15.

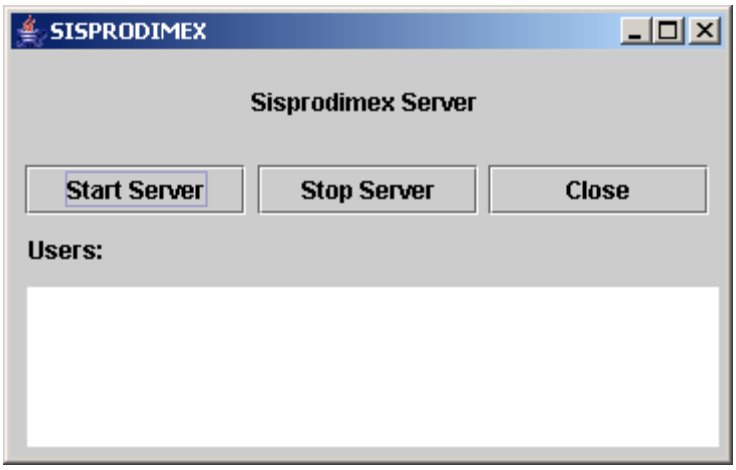

Figura 15 – Tela do Sisprodimex Server

A interface é constituída de três JButtons (*Start Server*, *Stop Server*, *Close*) e um JTextArea o qual é utilizado para visualização dos usuários conectados no servidor. Quando o JButton Start Server é pressionado, o método Start é chamado. Este método instância um objeto da classe ServerSocket para estabelecer a conexão.

```
server = new ServerSocket(DEFAULT_PORT);
```

O comando `socket = server.accept( );` diz ao programa que espere indefinidamente até que um cliente se conecte. Quando um cliente estabelecer uma conexão bem sucedida, esse método retorna um objeto `Socket` que representa a conexão feita. Tudo que o servidor envia para o fluxo de saída se torna a entrada do programa cliente e toda saída do programa cliente acaba no fluxo de entrada do servidor. Depois de estabelecida a conexão

com o cliente, o servidor cria um objeto **Thread** e o inicializa com o aplicativo SisprodimexServer **Runnable,** e depois chama o método **Start** da *thread.*

```
SisprodimexServer handler = new
SisprodimexServer(socket);
Thread thread = new Thread(handler);
thread.start();
```

O método **run** é sobrescrito chamando o método **handleRequest**. O método **handleRequest** é responsável por processar a solicitação do cliente. Ele recebe a solicitação do cliente, acessa a um banco de dados para a busca do arquivo solicitado e responde para o cliente da maneira mais conveniente. Ele cria três objetos, um **File,** um **FileInputStream** e um **DataOutputStream**.

```
File fp = new File(fileName);
FileInputStream fins = new FileInputStream(fp);
DataOutputStream o = new
DataOutputStream(request.getOutputStream());
```

O método **mostra** é responsável por configurar a GUI mostrada ao usuário.

Para testar a conexão com o *cluster* foi desenvolvida uma aplicação em Java, em um trabalho de mestrado correlato a este (PRIMO, 2005), utilizando os conceitos de JMPI-PLUS, para efetuar a consulta no banco de dados do SISPRODIMEX de um objeto no padrão DICOM e submetê-lo ao processamento no *cluster*, obtendo-se o tempo resultante deste processo. Os testes foram realizados com três tamanhos diferentes de imagens.

A figura 16 mostra o cluster experimental, cuja implementação foi baseado na arquitetura Beowulf e construído a partir de máquinas e dispositivos de baixo custo facilmente encontrados no mercado (PRIMO, 2005).

Figura 16 – Cluster baseado na Arquitetura Beowulf utilizado no projeto (PRIMO, 2005).

O gráfico da Figura 17 ilustra o resultado obtido com a simulação realizada entre o processamento de um objeto DICOM utilizando a implementação do JMPI-PLUS e uma imagem no formato JPEG utilizando a implementação do MPICH. Notou-se que a implementação do JMPI-PLUS obteve um melhor desempenho, pois utiliza a serialização de objetos (PRIMO, 2005).

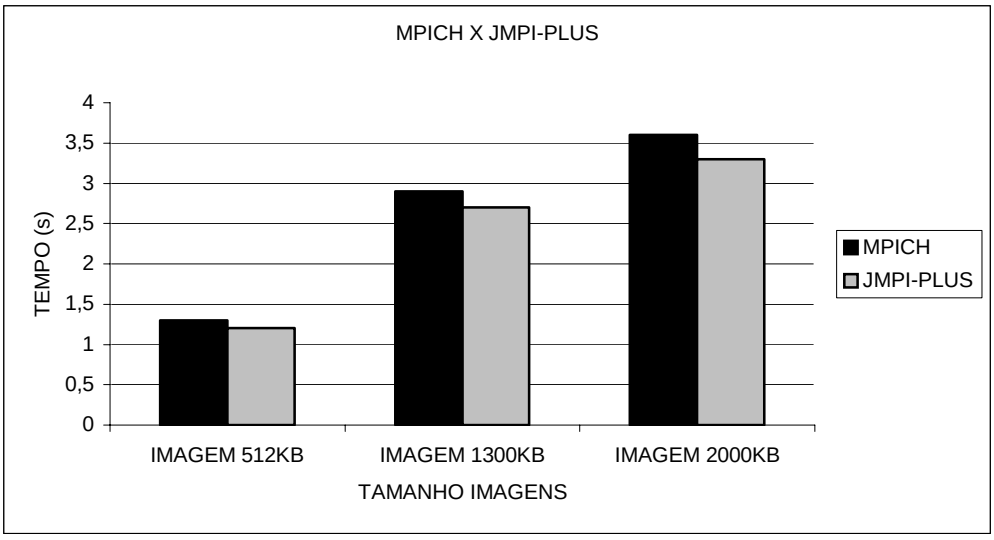

Figura 17 - Gráfico da simulação de processamento entre figura JPG e no padrão DICOM.

Outro objetivo deste teste, além dos experimentos com o *cluster*, foi também de validar o uso da linguagem Java para implementação das soluções expostas neste trabalho visto que a versão MPI proposta pelo JMPI-PLUS é baseada em na linguagem Java.

A conexão com o ambiente de *cluster* procura demonstrar também como a plataforma SISPRODIMEX pode ser extensível, agregando recursos propostos por outros trabalhos como já foi exposto.

Pode-se citar, como uma falta na coleta de resultados, testes com usuários finais (médicos, por exemplo) para avaliar a facilidade de uso da interface proposta, pois toda implementação foi baseada em solucionar os principais problemas encontrados e que foram os motivadores para a concepção do SISPRODIMEX. Portanto, este trabalho procura demonstrar os meios encontrados para resolver estes problemas.

Durante o processo de pesquisa e levantamento de referências bibliográficas não foram encontrados sistemas de gerenciamento de informações e imagens médicas que utilizassem um sistema baseado em *cluster* para obtenção de desempenho e tivessem a preocupação em utilizar tecnologias de (relativo) baixo custo. A maioria dos trabalhos encontrados concentram seu foco em decompor objetos DICOM em sistemas gerenciadores de banco de dados a fim de facilitar o acesso a imagens e informações médicas e criação de mecanismos para consulta de informações textuais através da Web. Na maioria dos casos, tais soluções procuram resolver problemas particulares de um ambiente sem a preocupação de interoperabilidade com ambientes externos.

CONCLUSÕES

Com o avanço tecnológico na área médica, os centros médicos, clínicos e hospitalares tradicionalistas, aos poucos, começam a ceder, frente aos benefícios e avanços dos sistemas PACS e do padrão DICOM ou daquilo que vêm sendo chamado de *Filmless Hospital*.

Este trabalho aborda a importância da democratização destas tecnologias a centros médicos de todos os portes, principalmente, os pequenos e médios, usando tecnologias abertas e de relativo baixo custo. Outros pontos relevantes são o uso de padrões para área de imagens médicas, como o DICOM, e o uso de Java e XML para a criação de soluções médicas e hospitalares em ambientes distribuídos, ressaltando para isso o uso de Web Services.

O SISPRODIMEX vem suprir a falta de uma ferramenta que se proponha a permitir que centros hospitalares de todos os portes possam também tirar proveito destes benefícios tecnológicos. É bem verdade que já existem algumas soluções no sentido de ampliar o raio de ação de sistemas PACS como os projetos que estão em constante desenvolvimento no InCor. Este, porém, tem como finalidade atender às necessidades locais dos sistemas PACS e a problemas particularmente ligados às necessidades de pesquisa do InCor e que, vez ou outra, acabam favorecendo a outros centros de pesquisa.

Uma outra grande preocupação é o montante de informações médicas geradas nos mais diversos centros médicos, principalmente, em ambientes PACS. Devido a tais informações estarem geograficamente dispersas, tal fato gera uma enorme perda em termos de informações para pesquisa médica. A partir da pesquisa realizada percebe-se a grande necessidade da criação de um grande banco de dados a fim de agregar tais informações. O

SISPRODIMEX resolve parcialmente esta questão, pois otimiza o uso de objetos DICOM organizando-os através de um sistema gerenciador de banco de dados.

A principio, o objetivo do SISPRODIMEX é fornecer o acesso às informações obtidas com as tecnologias PACS e DICOM a centros médicos e hospitalares.

Podem ser citadas algumas vantagens iniciais:

- Promover a comunicação de informações e imagens em formato digital independentemente dos fabricantes dos equipamentos.
- Permitir a criação e expansão de sistemas PACS entre os diversos sistemas de informação hospitalar.
- Permitir que a criação de uma base centralizada de informações médicas com independência de localização geográfica.

Podem ser citados também, alguns problemas que foram detectados e que dependem do avanço tecnológico ou estudos e pesquisas específicas para serem solucionadas, como:

- Por utilizar o sistema de imagens baseadas em pixels geram arquivos de grande volume e, conseqüentemente, exige grandes áreas de armazenamento.
- Gargalo no sistema de comunicação, pois, exige rede de alto desempenho.

A criação de padrões como o DICOM tem relevada importância quando observados pela ótica da criação de uma grande rede para trabalho colaborativo de pesquisa e diagnóstico sem que haja a preocupação da localização geográfica. Assim, o DICOM descreve os mecanismos para a existência de troca de informações entre os diversos equipamentos de modalidade.

Devido ao fato de Java e XML permitirem a criação de aplicações portáveis com uma certa facilidade, estes, naturalmente, tornam-se boas opções para o desenvolvimento de aplicações médicas, clínicas e hospitalares já considerando sua característica de ambiente heterogêneo e distribuído.

As implementações e testes iniciais com *cluster*s e o uso da metodologia JMPI-PLUS vêm a ratificar que a proposta deste trabalho, e uma vez que o ambiente se encontre completamente integrado a outros trabalhos, trará enormes contribuições tanto para a área de pesquisa baseada em informações e imagens médicas advindas de ambientes PACS quanto para as instituições que tem a nítida necessidade de obter um melhor gerenciamento e aproveitamento de suas informações.

Talvez, o fator mais importante do SISPRODIMEX, seja a proposta de centralização de informações médicas a partir de uma plataforma de arquitetura neutra e de relativo baixo custo que propõe oferecer alto poder de processamento. Um outro grande elemento motivador é o cansativo trabalho de busca de informações para pesquisas médicas, as quais, fundamentam a prática médica baseada em evidências. Então, supondo que todos os registros estejam disponíveis em uma rede ou na Internet, esta tarefa se tornaria mais fácil e até mais agradável, podendo-se obter resultados de pesquisas em tempo muito reduzido e analisar os resultados das mais diversas formas possíveis. No futuro, poderá ser comum um trabalho em conjunto entre grupos de pesquisa médica situados em localidades geográficas distintas. Esta é a idéia base para a implementação de gerenciamento de contexto.

Para isso, a Web Semântica propõe-se a melhorar a organização das informações na forma de dados estruturados, possibilitando buscas mais eficientes por agentes de software, pois, as pessoas consomem muito tempo para localizar uma informação na Web devido a ambigüidade de palavras-chave fornecidas pelas ferramentas de busca.

Na Web Semântica, um conceito importante está associado a uma URI (Uniform Resource Identifier), cuja principal função é atrelar um significado único e que qualquer pessoa poderá encontrar na Web. Para isso, são usadas Ontologias, cuja função é de fornecer o vocabulário necessário para a comunicação entre os agentes de software e as páginas Web e mostrar as relações entre os conceitos. Pode-se dizer, então, que uma ontologia é um conjunto

de documentos contendo a definição formal dos relacionamentos entre os conceitos destes documentos.

TRABALHOS FUTUROS

Como trabalhos futuros pode-se citar:

- Criação das conexões com outros projetos agregando maiores funcionalidades e tornando a plataforma extensível.
- Construção do *Data-Center* e *Data Warehouse* visando à composição de uma base inteligente de dados.
- Implementação de Web Semântica e Ontologias para área médica.
- Desenvolvimento do módulo de gerenciamento de contexto.
- Testes com cluster de maior poder de processamento.
- Testes junto a usuários finais.
- Testes com demais tipos de dados e imagens.
- Busca por desempenho na camada de transporte.

REFERÊNCIAS

AMDAHL, G. **Validity of the Single Processor Approach to Achieving Large Scale Computing Capabilities** – AFIPS Conference Proceeding. Washington DC: Thompson Books, Abril 1967.

ANDRIOLI, K.P. **Addressing the Coming Radiology Crisis: The Society for Computer Applications in Radiology Transforming the Radiological Interpretation Process (TRIP™) Initiative**, Novembro 2003, acessível em: <http://www.scarnet.org/pdf/TRIPwhitepaper1103.pdf>. Acesso em: 20 nov. 2004.

ARRUDA, L.; BAPTISTA, C.S.; LIMA, C.A. **MEDIWeb: A Mediator-based Environment for Data Integration on the Web**. In Proceedings of Fourth International Conference on Enterprise Information Systems, Cuidad-Real, Spain, Abril 2002.

BARRÉ, S. **Medical Image Samples**. Disponível em: <http://www.barre.nom.fr/medical/samples/>. Acesso em: 15 mar. 2004.

BERNERS-LEE, T.; HENDLER, J.; HENDLER, L. **The Semantic Web**. Disponível em: <http://www.sciam.com/article.cfm?articleID=00048144-10D2-1C70-84A9809EC588EF21>. Acesso em: 23 jan. 2004.

BERTOZZO JUNIOR, N.; FURUIE, S.S. **A Simple AVI-to-DICOM Converter in Angiography Images for PACS Applications**. Computers in Cardiology 98. In: IEEE Computers in Cardiology'98. Cleveland, EUA. 1998.

BERTOZZO JUNIOR, N.; FURUIE, S.S. **Sistema de Armazenamento e Distribuição de Imagens Médicas no PACS InCor**. Unidade de Pesquisa e Desenvolvimento – Serviços de Informática. Instituto do Coração – FMUSP. Disponível em: <http://www.informedicajournal.org/a1n2/files/resultados.php?evento=4>. Acesso em: 18 nov. 2003.

BRAY, T; Paoli, J.; Sperberg-McQueen, C.M. **Extensible Markup Language – XML 1.0** – Second Edition. Outubro 2000.

COHEN, S. **PACS and Electronic Health Records**. IBM Haifa Research Labs.Haifa University, Monte Carmel, Israel. 2004.

CORMACK, A.M. **Early two-dimensional Reconstruction and Recent Topics Stemming from it.** Physics Department, Tuffs University, Medford, Mass, USA. Nobel Lecture. 8

dezembro de 1979. Disponível: <http://www.nobel.se/medicine/laureates/1979/cormack-lecture.html>. Acesso em: 23 nov. 2003.

CORREIA, V.M**. Serviços de Web-Semântica para Transporte de Imagens Médicas**. Relatório Anual de Projeto de Iniciação Científica. Processo FAPESP Nº 03/07557-9. Período de Outubro/2003 a Julho/2004.

DECKER, S.; MELNIK, F**. The Semantic Web: The Roles of XML and RDF**. IEEE Internet Computing. Setembro-Outubro 2000.

DEITEL, H.M.; DEITEL, P.J.; NIETO, T**. Internet & World Wide Web – Como Programar**. 2. Porto Alegre, RS, Brasil: Ed. Bookman, 2003A.

DEITEL, H.M.; DEITEL, P.J**. Java – Como Programar**. 4. Porto Alegre, RS, Brasil: Ed. Bookman, 2003B.

DEITEL, H.M.; DEITEL, P.J.; NIETO, T. **XML – Como Programar**. Porto Alegre, RS, Brasil: Ed. Bookman, 2003C.

FERGUSON, D.F.; STOREY, T.; LOVERING, B. **Secure, Reliable, Transacted Web Services: Architecture and Composition**. White-paper: IBM Corporation & Microsoft Corporation. Disponível em: <http://www-306.ibm.com/software/solutions/Webservices/pdf/SecureReliableTransactedWSAction.pdf>. Acesso em: 25/jan/2004.

FIREBIRD. **Relational Database for the New Millenium**. Disponível em: <http://firebird.sourceforge.net/>. Acesso em: 25 jan. 2004.

GREENES, R.A.; BRINKLEY, J.F. **Radiology Systems. In: SHORTLIFE, E.H.; PERREAULT, L.E. (EDS.). Medical Informatics**. USA: Addison-Wesley, 1990. p.324-365.

GRUBBER, T.R. **Translation Approach to Portable Ontologies**. Journal on Knowledge aquisition, Vol. 5(2), 199-220, 1993.

GUARINO, N. **Semantic Matching: Formal Ontological Distinctions for Information Organization, Extraction and Integration**. Berlin: Springer Verlag, p.139-170, 1997.

HAMILTON, M.A. **Java and The Shift to Net-centric Computing**. IEEE Computer, 29, pp. 31-39, Agosto, 1996.

HENDLER, J. **Agents and the Semantic Web. IEEE Intelligent Systems**, March-April 2001, Vol. 16, No. 2, pp 30-37. Disponível em: <http://csdl.computer.org/comp/mags/ex/2001/02/x2toc.htm>. Acesso: 23 jan. 2004.

HOUNSFIELD, G.N. **Computed Medical Imaging**. The Medical Systems Departament of Central Research Laboratories EMI, London, England. Nobel Lecture. 8 dezembro de 1979. Disponível: <http://www.nobel.se/medicine/laureates/1979/hounsfield-lecture.html>. Acesso em: 23 nov. 2003.

HUANG, H.K. **PACS – Basic Principles and Application**, Wiley-Liss, New York, 1999.

IBEXPERT. **Site oficial do IBExpert**. Disponível em: <http://www.ibexpert.com> Acesso em: 15 mar. 2005.

IDRIS, N. **Should I use SAX or DOM?** Maio, 1999. Disponível em: <http://developerlife.com/saxvsdom/default.htm>. Acesso em: 25 jan. 2004.

IMAGEJ. **Site oficial do ImageJ: ImageJ Processing and Analisys in Java**. Disponível em: <http://rsb.info.nih.gov/ij/>. Acesso em: 25 jan. 2004.

KREGER, H. **Web Services: Conceptual Architecture (WSCA 1.0) – IBM Software Group 2001**. Disponível em: <http://www-3.ibm.com/software/%20solutions/webservices/pdf/WSCA.pdf>. Acesso em: 15 dez. 2004.

MARQUES, P.M.A.; SANTOS, A.C.; ELIAS, J. **Implementação de um Sistema de Informação em Radiologia em Hospital Universitário**. Radiol Bras 2000. 33:155-160.

MCCARTH, S. XML – **Aplicações Práticas – Como Desenvolver Aplicações de Comércio Eletrônico**. Editora Campus, 1999.

MELLO, R.S.; DORNELES, C.F.; KADE, A. **Dados Semi-Estruturados**. In: XV Simpósio Brasileiro de Banco de Dados. João Pessoa, Paraíba, 2000.

MORENO, R.A. **Utilização de Contexto para Visualização de Imagens Médicas**. Anais do Congresso Brasileiro de Informática na Saúde – CBIS'2004, Ribeirão Preto, Novembro de 2004.

MUCHERONI, M.L.; TAMAE, R.Y. **SISPRODIMEX – Um Sistema Distribuído de Imagens Médicas**. Anais do Congresso Brasileiro de Informática na Saúde – CBIS'2004, Ribeirão Preto, Novembro de 2004.

NETBEANS. Site oficial NetBeans IDE. Disponível em: <www.**netbeans**.org>. Acesso em: 20 jan. 2003.

NOBEL. **Nobel e-Museum**. Wilhelm Conrad Rõntgen Biography. Disponível em: <www.nobel.se/physics/laureates/1901/rontgen-bio.html>. Acesso em: 18 nov. 2003.

PACHECO, P.S. **Parallel Programming with MPI**. Library of Congress Cataloging-in-Publication Data. San Franciso, CA, USA: Morgan Kaufmann Publishers, 1997.

PERRY, P.J. **Creating Cool Web Applets with Java**. IDG Book Worldwide, ISBN 1-56884-881-1, 1996.

PRIMO, A.L.G. **Serialização e Performance em Ambientes Distribuídos Usando MPI**. 2005. Dissertação (Mestrado em Ciência da Computação) – Programa de Pós-graduação em Ciência da Computação (PPGCC) Centro Universitário Eurípides de Marília – UNIVEM. 2005.

RASBAND, W. **Research Services Branch**, National Institute of Mental Health, Bethesda, Mariland, USA. Disponível em: <http://rsb.info.nih.gov/>. Acesso em: 15 mar. 2005.

SANTOS, M.; RUIZ, E.E.S. **Desenvolvimento de Aplicações DICOM com Tecnologias Web: Um servidor e cliente DICOM**. In: CBIS'2002 - VIII Congresso Brasileiro de Informática em Saúde – Natal-RN, Setembro-Outubro de 2002.

SOUZA, G.P.; PFITSCHER, H.; MELO, A.C.M.A. **Computação Distribuída Baseada em Java Rodando em Arquiteturas Beowulf e Arquiteturas Heterogêneas**. Departamento de Computação – Universidade de Brasília (UNB). 2003.

SHIRALKAR, S.; LEWIS, M. **Filmless Hospital: How Its Affect Trainees?**. Royal Glamorgan Hospital, South Wales. Pp. 104-106. 2003.

SIEGEL, E.L. **Current State of Art and Future Trends**, in Filmless Radiology. E.L. Siegel, R.M. Kolodner. New York City, NY, USA: Springer Verlag. Pp. 3-20. 1999A.

SIEGEL, E.L.; KOLODNER, R.M. **Filmless Radiology**. New York City, NY, USA: Spring Verlag, 1999B.

STAAB, S.; ANGELE, J**. Semantic Community Web Portals**. University Germany, 2000.

SUN. **The Java Programming Language**. Sun Microsystems S/A, 2001.

SUN. **Site oficial do Java**: <http:// www.java.sun.com>. Sun Microsystems S/A, 2002.

TIANI. **Site oficial da Tiani MedGraph**. Disponível em: <http://www.tiani.com>. Acesso em: 15 mar. 2005.

TORRES, G.L., COSTA, C.I.A. **Advances in Intelligent Systems and Robotics: A Paraconsistent Inference Engine for The Semantic Web**, IOS Press, Amsterdam, Netherlands V.101, p.51-58. 2003.

VELOSO, R.R. **Java e XML – Processamento de Documentos XML com Java**. Editora Novatec, 2003.

WALKER, B.J. **Introduction to Single System Image Clustering**. Disponível em: <http://www.sourceforge.net.>. 2001.

W3C-RDF. **RDF – Resource Description Framework**. Disponível em: <http://www.w3.org/RDF/>. Acesso em: 23 jan. 2004.

W3C-XML. **XML – Extensible Markup Language**. Disponível em: < http://www.w3.org/XML/>. Acesso em: 23 jan. 2004.

WHITE-SAND. **Introdução ao Padrão DICOM**. In: V Treinamento de Padrão DICOM de Imagens Médicas. São Paulo, SP. 11-12 Jul. 2003.

Livros Grátis

(http://www.livrosgratis.com.br)

Milhares de Livros para Download:

Baixar livros de Literatura
Baixar livros de Literatura de Cordel
Baixar livros de Literatura Infantil
Baixar livros de Matemática
Baixar livros de Medicina
Baixar livros de Medicina Veterinária
Baixar livros de Meio Ambiente
Baixar livros de Meteorologia
Baixar Monografias e TCC
Baixar livros Multidisciplinar
Baixar livros de Música
Baixar livros de Psicologia
Baixar livros de Química
Baixar livros de Saúde Coletiva
Baixar livros de Serviço Social
Baixar livros de Sociologia
Baixar livros de Teologia
Baixar livros de Trabalho
Baixar livros de Turismo